COMX—PC1 电脑实验与指导

焦民 刘照雄 编著

宇航出版社

COMX—PC1电脑实验与指导／焦民，刘照雄编著．—北京：宇航出版社，1985.8
198页；32开
1.50元

证号	姓名	日期	证号	姓名	日期	证号	姓名	日期
384400		09						

COMX—PC 1
电脑实验与指导

焦 民 刘照雄 编 著

宇航出版社

内 容 简 介

本书以教育部推广的普及型电脑 COMX—PC1 为对象，全面系统地介绍了电脑系统的设备、构成、上机操作等基础知识及BASIC语言；结合实例介绍了该电脑的编辑、音响、趣味等功能；并在附录中全面介绍了该电脑外部设备的使用方法及系统的详细技术资料。

本书文字浅显，内容通俗易懂，特别注重上机操作与实践。书中提供的程序紧密结合中学生所学的课堂知识及生活，生动有趣，引人入胜。本书可作为中学计算机正规教学及开展计算机课外活动的参考教材。对于具有中等文化水平而又迫切希望掌握电脑的使用方法和学习编程的广大自学者来说（包括城镇， 农村的个体户，专业户），本书还可作为速成学习电脑的辅导教材。

COMX—PC1

电脑实验与指导

焦民　刘照雄　编著

＊

宇航出版社出版

新华书店北京发行所发行

各地新华书店经售

保定振兴包装印刷厂印刷

＊

开本：787×1092 1/32 印张：6 7/16 字数：144千字

1985年8月第1版 1985年8月第1次印刷

印数：1—15,000册 定价：1.50元

前　言

COMX—PC1 电脑已在我国部分中学试用一年多了。通过一年多的教学与实践活动，根据COMX—PC1（COMX—35的改进型，二者的软、硬件全部兼容）电脑的特点，我们编写了这本紧密结合上机操作与实践的电脑实验指导。目的在于为使用 COMX—PC1 电脑的教师、学生学习和应用电脑提供参考。

全书共分十章，第一章介绍 COMX—PC1 电脑系统设备及构成；第二章介绍 COMX—PC1 电脑的使用及键盘操作；第三章介绍 COMX—PC1 电脑作为计算器的使用方法；第四章至第十章介绍COMX BASIC程序设计语言。附录中全面介绍了 COMX—PC1 电脑的外部设备使用方法及系统的详细技术资料。

本书文字浅显，内容通俗易懂，注重上机实践。对于电脑选修课的上机实习，是一本不可缺少的指导书。全部内容可以安排十五课时上机：前三章上机一次；第四章上机一次；第五章上机一次；第六章上机二次；第七章上机一次；第八章上机三次；第九章上机二次；第十章上机二次；附录部分上机一次。

本书可作为中学计算机正规教学及开展计算机课外活动的参考教材。对于具有中等文化水平而又迫切希望掌握电脑的使用方法和学习编程的广大自学者来说，本书还可以作为速成学习电脑的辅导教材。如果读者身边有一台COMX—

PC1电脑，边读本书边上机实践，很快就能初步掌握COMX—PC1电脑的操作并可以编写一些简单的程序。

我们热切希望本书能帮助COMX—PC1电脑的用户尽快学会使用COMX—PC1电脑及BASIC语言，并在此基础上继续提高。若本书能为电脑在我国的普及应用有所助益，我们将感到欣慰。

由于我们水平有限，实践经验也不够丰富，缺点错误在所难免，恳请广大读者批评指正。

编　　者

1984.12.

目　　录

第一章 COMX—PC1 电脑系统简介

§1 COMX—PC1电脑系统的设备

COMX—PC1 电脑是一种技术先进、用途广泛、易于掌握的普及型电脑。其独特的趣味功能，使它成为青少年学习电脑的好友。

COMX—PC1 电脑系统同其它类型的电脑一样，由运算器、控制器、存储器以及输入、输出设备组成。

电脑的运算器、控制器是电脑的中心部件。特别是控制器，它是电脑的“神经中枢”，由它统一指挥和控制电脑各个部分协调工作。一般把这两个主要部件称为CPU（中央处理器）。COMX—PC1电脑的CPU型号是1802A，它是一种集成度很高的芯片，即所谓的CMOS。1802A在处理数据时，一次可处理八位二进制数，因而称COMX—PC1电脑为八位机。

COMX—PC1电脑的存储器是用来保存大量信息的设备。它们分为内存储器和外存储器。内存储器简称内存，是由大规模集成电路芯片充任。它存取速度快，直接与PCU发生联系。内存一般用来储存程序和数据。

存储器的容量通常用存储单元的多少来衡量。一个存储单元里存放的一组二进制数叫一个字。一个字所包含的二进制数的位数叫字长。COMX—PC1电脑存储器的字长是八

位。

内存还分为只读存储器（ROM）和随机读取存储器（RAM）。COMX—PC1电脑的ROM容量为16K（1K相当1024个存储单元），它里边存有COMX BASIC解释程序。电脑靠这个解释程序才能进行工作。COMX—PC1电脑的RAM容量为35K，其中32K提供给用户使用。可以存储程序和数据。

由于COMX—PC1电脑内存容量有限，特别是当电源断开后，内存中的信息将全部丢失。这样就需要有外存储器。外存储器容量大，可永久保存信息。COMX—PC1电脑的外存储设备最常用的是一般盒式磁带录音机。在高质量的磁带上可以录上所需要的各种信息。比如，游戏程序、教学程序、数据等。为了使COMX—PC1电脑有更强的功能，还配备了软磁盘机。由于软磁盘机操作复杂且价格较贵，一般不使用它。

外部设备也就是输入、输出设备。COMX—PC1电脑的输入设备是键盘，输出设备是显示器（监视器或电视机）、功能扩展接口板和打印机。它们是电脑同外部发生联系（比如同人发生联系）的媒介，是电脑系统必不可少的设备。

COMX—PC1电脑的显示器可以是监视器。不论是单绿色的还是彩色的，也不论是12英寸的还是14英寸的，都可以由视频输入接口通过视频信号线与电脑沟通联系。此外，普通家用电视机也可作为电脑的显示器。不论是黑白接收机还是彩色接收机，也不论是PAL制的还是NTSC制的均可充任。由接收机的天线接口通过电视信号连线沟通与电脑的联系，可以在电视屏幕上得到满意的图形和文字。

有时，我们需要将一些结果打印在纸上。COMX—PC1电脑的配套功能扩展接口板及打印机提供永久保留信息的条件。

COMX—PC1电脑功能扩展接口板是一块由多个集成电路芯片组成的附属部件。它的加入使电脑功能倍增。它不但能使COMX 35P打印机打输出印信息，而且可使用户直接用机器码来利用电脑，为机器效率的提高创造了条件。

COMX 35P打印机是COMX—PC1电脑最理想的配套信息记录装置。用该机打印的字符、图形美观清晰。

COMX—PC1电脑使用的键盘与CPU、RAM、ROM及外设接口一起构成COMX—PC1主机。

§2 COMX—PC1电脑系统构成

根据用户使用电脑的目的不同，系统大致可分为以下三类。

一、基本系统

由主机、显示器组成基本系统。利用基本系统可以进行全部非保留性程序设计工作，可供BASIC语言程序设计使用。该系统建立步骤如下：

1．将电视信号线接至电视机的外接天线接口和电脑的输出接口（VHF接口）。如果使用监视器则应将视频信号线接监视器输入接口和电脑的视频输出接口。

2．将主机所带电源变压器次级小插头插入主机电源接口（POWER）。

3．将电视机或监视器电源及主机电源变压器接到交流

202V电源上。（见图1.1）

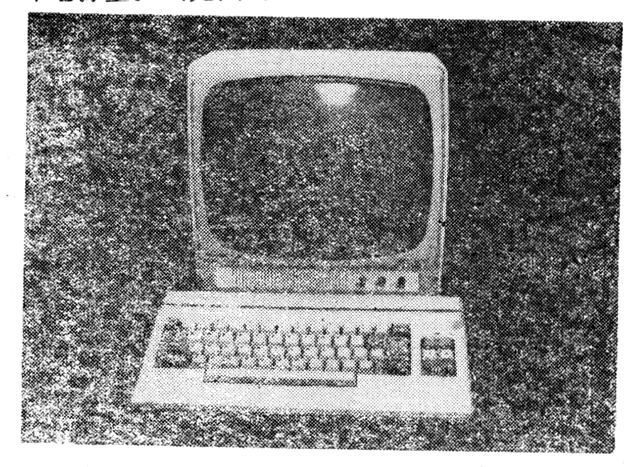


图 1: 1 COMX－PC1 电脑基本系统

二、扩充系统

由基本系统、外存储装置（录音机）组成扩充系统。扩充系统除了具有基本系统的全部功能外，还可以进行程序、数据的存储、读取，文件的建立等工作。该系统的建立步骤如下：

1．同基本系统建立步骤1、步骤2。

2．将主机所带红色插头线的一头接主机CASSETTE EAR 接口，另一头接录音机的 EAR 或 EARPHONE 或 MON 或 MONITOR 接口均可。将黑色插头线的一头接主机CASSETTE MIC 接口，另一头接录音的 MIC 或 MICROPHONE 接口。

3．将电视机或监视器、录音机、主机电源变压器接到交流220V电源上。（见图1.2）

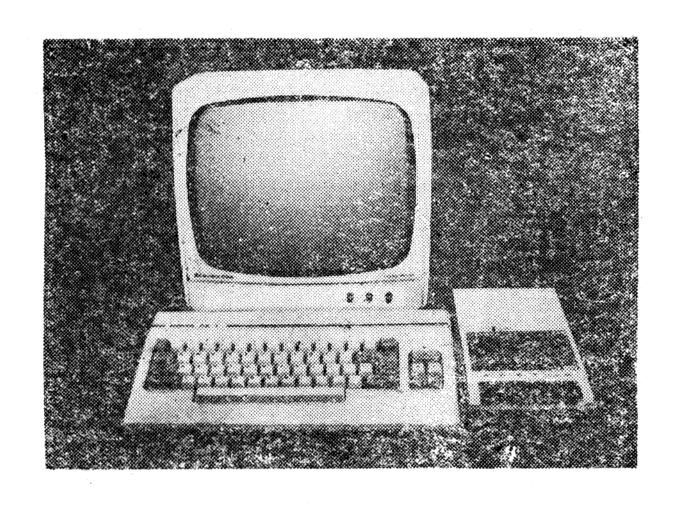


图 1.2 COMX－PC1电脑扩充系统

三、全系统

由全部设备组成全系统。该系统除具有扩充系统的全部功能外，还具有打印、监控、编写机器语言程序等多种用途。该系统建立步骤如下：

1．同基本系统步骤1、2。

2．同扩充系统步骤2。

3．将功能扩展接口板插入主机右侧的插座内，用电缆将功能扩展接口板与COMX 35P打印机接好。

4．将COMX 35P打印机所带电源变压器的小插头插入打印机电源插口。

5．将全部设备的电源插头接到交流220V电源上。（见图1.3）

通常，录音机可随时接入或取出，即使是在电脑工作

时，也不会产生不良影响。而功能扩展接口板则必须在全部设备断电情况下插入或拔出，否则，将造成功能扩展接口板的损坏，影响电脑的使用。

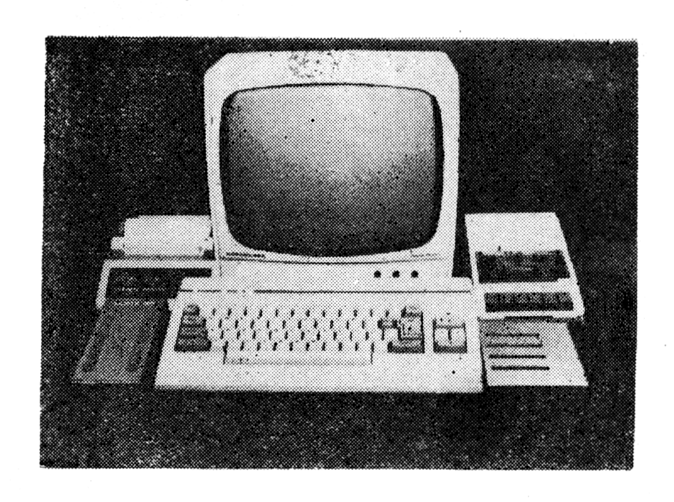


图 1.3 COMX—PC1电脑全系统

COMX—PC1电脑基本系统的使用在正文中予以介绍。扩充系统及全系统的使用在书末的附录中介绍。

第二章　基本操作和键盘

§1　基本操作

一、开机

在基本系统构成后，可按以下步骤操作。

1．将主机电源开关置OFF（关）位置。

2．将电视机的频道开关置二频道。（这里以 PAL 制电视机为例。如果使用NTSC制电视机请参看附录。使用监视器不需要调频道。）

3．开启总电源。

4．开启电视机电源。

5．置主机电源开关于 ON 位置。此时可以听到一支悦耳的小曲子，主机上的红色指示灯亮了。

6．调整电视机频道微调器，直到屏幕上出现

C O M X

或

C O M X

COPYRIGHTED © 1983 BY

COMX

WORLD OPERATIONS LTD

信息。

7．调整电视机的亮度、对比度等直到满意为止。

8．按键盘上的任一个键（空格棒除外），屏幕上会出现

COMX BASIC V1.00
READY
：♦

这时，电脑系统已经建立，并且处于 BASIC 控制下，可以利用BASIC语言工作了。

“：”为COMX BASIC 的提示符。“♦”称为光标。当屏幕上出现“：♦”信息时，就可以使用BASIC指令进行工作了。

二、关机

关机操作步骤：

1．置主机电源开关于OFF位置。

2．关电视机电源开关。

3．关掉总电源。

§2 COMX—PC1电脑键盘

键盘是我们与电脑进行对话的工具。熟悉每一个键的功能，熟练掌握键盘操作会大大提高使用电脑的效率。

一、键盘

COMX—PC1电脑键盘见图2.1。

键盘上共有60个键。它们包括：

1．大写英文字母键 A，B，……X，Y，Z

2．数字键 0，1，2，3，4，5，6，7，8，9

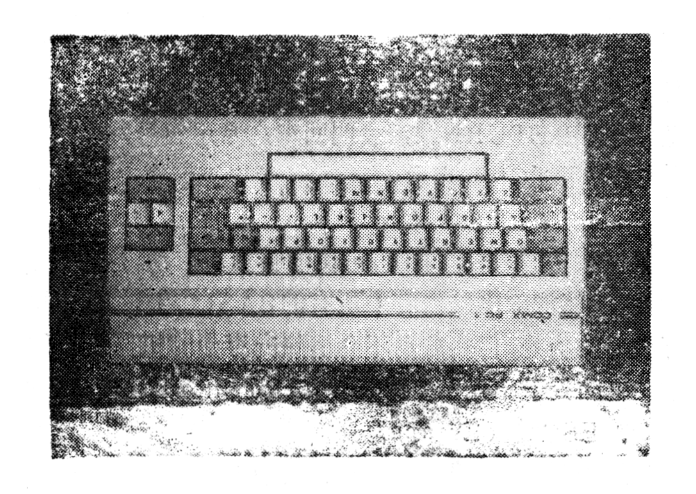


图 2.1 COMX－PC1电脑键盘

3．符号键　＋，－，＊，／，∧等

4．功能键　CTRL，SHIFT，ESC，RT，RTL等

5．空格棒

6．方向控制键　上移（▲），左移（◀），下移（▼），右移（▶）。

二、键

键盘上的字母键、数字键的排列同普通英文打字机键盘一致。其它的键排列各有不同。下面简单地介绍一些键的功能。

1．CR键（Carriage　Return）

CR键可以使光标移至下一行的最左边，并且使电脑接受指令。在键盘上按键后，屏幕上会出现输入的信息，但此

时电脑内存并未接受任何信息，因而CR键实际上具有使电脑接受指令的功能。此外，当输入信息发生错误或屏幕上光标不在最左边时，按一下CR键可以使电脑处于常态。CR键通常也叫RETURN键，或叫回车，常用符号＞表示。本书的CR键用〔CR〕表示。

2．RT、RTL键（Reset）

同时按RT键与RTL键，可使电脑重新启动。这相当于热启动。此时将清除电脑内存中所有的BASIC程序，使电脑回到开机时的状态。但热启动不会清除内存中的机器语言程序。通常在电脑处于“死”机时，同时按RT和RTL键重新启动。

3．ESC键（Escape）

ESC键按下后将中断正在执行中的BASIC指令。比如，在执行BASIC程序时，如果按ESC键，屏幕上会出现

```
ERR CODE 0
AT LINE n
READY
：◆
```

n为执行中断时的行号数。在中断后，电脑等待接受新的BASIC指令。

4．CTRL键（Control）

CTRL键单独按下不起任何作用，必须同其它键一起使用。CTRL键同其它键一起按下时，有一些特殊的功能，这将在以后专门介绍。本书以CTRL—X（X为其它键的名字，一般为字母键）表示。比如CTRL—I表示同时按下CTRL和I两个键。

5．SHIFT键（Shift）

SHIFT 键具有变换的功能，它必须同其它键同时按下才起作用。有些键的上方的符号就是 SHIFT 键与该键同时按下产生的。比如"（"、"∧"等都是要自身与 SHIFT 键同时使用。书中此类符号将直接写出。而SHIFT键与字母键同时按则用 SHIFT—X（X为字母键名）表示。比如 SHIFT—A 就表示 SHIFT 键和A键同时按下，SHIFT—M 表示 SHIFT 键和M键同时按下。

6．DEL键（Delete）

DEL键具有清除字符的功能。按DEL键就会使光标左移一格，并在屏幕上清除该字符，直至光标的左边为提示符时。DEL键才失去作用。

7．算术运算符键

\+ 加号键

\- 减号键

* 乘号键

／ 除号键

∧ 乘方号键

8．数字 0 和字母O

数字 0（零）和字母O很相似，因而按键时要特别注意。在屏幕上零显示为⊘（中间有一斜道）用来区别英文字母O。书中 0 没有划线，读者千万注意不要误认为是字母O。

§3 按键的基本指法和要领

一、基本位置

左手按下列键

A　　　　S　　　　D　　　　F　　　空格棒
（小指）　（无名指）（中指）　（食指）（拇指）

右手按下列键

空格棒　　J　　　K　　　L　　　；
（拇指）　（食指）（中指）（无名指）（小指）

二、手指按键分工

各手指按键的分工见图2.2

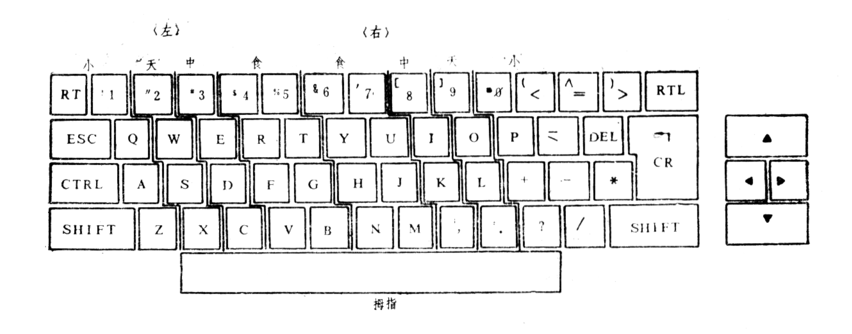


图 2. 2　手指按键分工

三、按键要领与技巧

按键时要迅速、准确、果断，要轻轻地按，不要按住键不放，只要感到已经按下就可以了。

按键时要逐步养成眼睛不看键盘，根据手指分工按键的好习惯，这样会提高输入字符的速度。

凡需同时按两个键时，例如 SHIFT—I 可以先按住 SHIFT 键，然后再按I键。这样可以保证同时按下两个键，从而避免出错。

§4 练 习 题

一、字母键练习

做练习时每打完一行要按一下CR键。如果按错了字符则要用DEL键改正。

根据要求按如下键

A S D F J K L I

ADJL SFK AFL DJSK

F G H J C V B N

AGCV LHNB SFGH JKVB

P Q O W Z M X

PQX MWXZ MOPW OWR

E R U I T Y

YER TUR RIU EYI

ABCDEFGHIJKLMNOPQRSTUVWXYZ

此时屏幕上已充满字符，如果想清除它们，不必关机，也不必按RT和RTL键，因为那样对机器毫无好处，只要执行以下操作即可：打入

CPOS（0，0）：CLS 〔CR〕

此时屏幕上就没有其它字符了。

二、常用字符、数字键练习

根据要求按键

1+2－3+4*5／6－7+8*9／10 〔CR〕

112*103+579－326 〔CR〕

：；，．< >=∧#$（）“ 〔CR〕

三、常用BASIC字练习

先将下面的BASIC程序输入到电脑中。

```
NEW    〔CR〕
5   CPOS (0, 0) : CLS                  〔CR〕
10  FOR I=1 TO 30                      〔CR〕
20  READ A$ (I)                        〔CR〕
30  NEXT I                             〔CR〕
40  DEFINT R                           〔CR〕
50  R=RND (29) + 1                     〔CR〕
60  CPOS (10, 15) : CLS                〔CR〕
70  CPOS (10, 15) : PRINT A$ (R)       〔CR〕
80  CPOS (15, 14) : CLS                〔CR〕
90  INPUT B$                           〔CR〕
100 IF B$=A$ (R) GOTO 50               〔CR〕
110 GOTO 80                            〔CR〕
200 DATA "LET","PRINT","INPUT"         〔CR〕
210 DATA "GOTO","IF","THEN",
         "REM"                         〔CR〕
220 DATA "READ","DATA",
         "RESTORE"                     〔CR〕
230 DATA "END","DIM","FOR","TO"        〔CR〕
240 DATA "NEXT","GOSUB",
         "RETURN"                      〔CR〕
250 DATA "LIST","NEW","RUN"            〔CR〕
260 DATA "STEP", "SIN", "COS"          〔CR〕
270 DATA "LOG","EXP","INT","SQR"       〔CR〕
```

280 DATA "SGN","ABS","TAB" 〔CR〕

输入完毕后打

LIST 〔CR〕

屏幕上会出现上述程序清单。检查是否有错。如果有错，只需重新打入该行即可（注意修改行的最左的数字应与原来的一致），无误后打

RUN 〔CR〕

此时，在屏幕中央会出现一个词，下方有个“？”，照出现的词打一遍，然后按CR键。如果操作正确就会出现另外一个词；如果操作失误，就会抹去你打的字，此时需重新打。要想结束这个练习，只需按ESC键，再按CR键就行了。停止以后再想运行，请再打

RUN 〔CR〕

即可。

第三章　COMX—PC1

电脑做为电子计算器的使用方法

一般的电脑都可以做为计算器使用。COMX—PC1电脑当然也不例外。但电脑当计算器使用时不象一般电子计算器只靠按功能键完成计算，还要借助PRINT指令来实现。

§1　一般计算

例 1.1　计算5＋6

打入

PRINT　5＋6　　〔CR〕

屏幕上会出现

11

在计算时，任何运算符号都不能省略，但在算式的后面一定不要打等号，而在算式的前面一定要使用PRINT指令。当然，CR键更是不能忘记按的。

例 1.2　计算下列各题

（1）75－17

打入

PRINT　75－17　　〔CR〕

结果是

58

（2） 7×8

```
PRINT  7 * 8                    〔CR〕
56
```

（3） $56 \div 8$

```
PRINT  56／8                    〔CR〕
7
```

（4） 2^5

```
PRINT 2 ∧ 5                     〔CR〕
32
```

小括号可以在算式中使用

例 1.3 计算下列各题

（1） $(5+6)(9-7)$

打入

```
PRINT  (5+6) * (9-7)            〔CR〕
```

结果是

```
22
```

注意乘号用“*”，不能打“·”、“×”，更不能省略。

（2） $(7-4)(19-14)/(8-3)$

打入

```
PRINT  (7-4)*(19-14)/(8-3)      〔CR〕
```

结果为

```
3
```

注意除号用“/”，不能用“÷”。

（3） $(3^3+2^4)/(2\times5)$

打入

PRINT（3∧3＋2∧4）／（2＊5）　【CR】

结果为

4.3

值得注意的是，分子、分母上的算式需分别加上小括号，否则答案就不对了。

§2　较复杂的计算

同其它电脑一样，COMX—PC1电脑内设立了一些基本数学函数和特殊函数。这些函数除了可以在程序中使用外，也可以象在多功能计算器中那样直接调用。表3.1列出10种基本函数和它们的功能。

表 3.1　　基本函数表

函数式	功能
SIN（X）	求x的正弦函数值　（X为弧度）
COS（X）	求x的余弦函数值　（x为弧度）
ATN（X）	求x的反正切函数值
EXP（X）	求指数函数e^x的值
LOG（X）	求自然对数函数的值
SQR（X）	求x的算术平方根值
ABS（X）	求x的绝对值
INT（X）	求不大于x的最大整数值
SGN（X）	求x的符号值
PI	求π值　（3.14159）

例 2.1　计算下列各题

（1）$|-30|$

```
PRINT ABS (-30)          〔CR〕
30
```

（2）arctg 1

```
PRINT ATN (1)            〔CR〕
.785399
```

（如果输出数值小于1，则小数点左边的零省略）

（3）e^2

```
PRINT EXP (2)            〔CR〕
7.38906
```

（4）ln10

```
PRINT LOG (10)           〔CR〕
2.30259
```

（5）sin30°

```
PRINT SIN (PI/6)         〔CR〕
.5
```

（6）cos60°

```
PRINT COS (PI/3)         〔CR〕
.500001
```

在计算时是有一定误差的。

（7）$\sqrt{65}$

```
PRINT SQR (65)           〔CR〕
8.06226
```

（8）对3.758取整

```
PRINT INT (3.758)        〔CR〕
```

3

（9）对－4.762取整

```
PRINT  INT (-4.762)            〔CR〕
- 5
```

（10）分别取－6.8，0，90的符号

```
PRINT  SGN (-6.8)              〔CR〕
- 1
PRINT  SGN (0)                 〔CR〕
0
PRINT  SGN (90)                〔CR〕
1
```

符号函数的取值规则是：

自变量大于0，函数值为1；

自变量等于0，函数值为0；

自变量小于0，函数值为－1。

通常，电脑的常态为弧度制。尽管机内设置了PI函数，但仍然觉得不方便。COMX—PC 1电脑提供了定义度制、弧度制的指令。具体做法是先打入

```
DEG                    〔CR〕
```

以后在计算三角函数时，自变量就可以用度为单位了。

例 2.2 计算sin30°

先打入

```
DEG                    〔CR〕
```

再打入

```
PRINT  SIN (30)                〔CR〕
.5
```

例 2.3 计算cos60°

可直接打

```
PRINT  COS(60)                [CR]
.5
```

要想使电脑再回到弧度制，只要打

```
RA                            [CR]
```

即可。

在计算函数值时，函数的自变量可以是常数，也可以是代数表达式，还可以是函数式。在计算时不要忘记自变量要放在小括号内。

例 2.4 计算sin (arctg 1)

直接打入

```
PRINT  SIN (ATN (1))          [CR]
```

结果为

```
.707103
```

§3 导出函数

COMX—PC 1 电脑中没有设置正切函数，但我们可以用sinx／cosx求出tgx。可以由基本函数导出的函数见表3.2。

表 3.2　　　　导出函数表

函　数　式	用基本函数表示式
tgx	SIN (X) /COS (X)
ctgx	COS (X) /SIN (X)
secx	1/COS (X)
cscx	1/SIN (X)
arcsinx	ATN (X/SQR (1 — X*X))
arccosx	-ATN (X/SQR (1-X*X)) +1.5708
arcctgx	-ATN (X) +1.5708
racsecx	ATN (SQR (X*X-1)) + (SGN (X) -1) *1.5708
arccscx	ATN (I/SQR (X*X-1)) + (SGN (X) -1 *1.5708
sinhx	(EXP (X) -EXP(-X)/2
coshx	(EXP (X) +EXP (-X)) / 2
tghx	-EXP (X)/(EXP (X) -EXP (-X))*2+1
arcsinhx	LOG (X+SQR (X*X+1))
arccoshx	LOG (X+SQR (X*X-1))
arctghx	LOG ((1+X)/(1-X))/2

§4　练　习　题

一、计算下列各题

（1）　求半径为2.5 cm的圆的面积。（19.635cm^2）

（2）　求边长为3.2 cm的正方体的体积。
（32.768cm^3）

（3）　求直径为7.1 cm的圆的周长。（22.3053cm）

（4）　1/5 + 1/4 + 1/3 + 1/2　　　　（1.28333）

（5）　7 − ×8 ÷ 6 − 3/2 + 1 − 4 + 5×7

(21.1667)

(6) $2^{10}-3^{4}\times5^{2}$ (−1001)

(7) 1 + 2 + 3 + 4 + 5 …… + 2 (210)

二、求下述函数值

(1) tg36° (.726543)

(2) ctg50° (.8391)

(3) sec20° (1.06418)

(4) csc40° (1.55573)

(5) cos294° (.406737)

(6) arcsin (0.5) (30°)

(7) arccos (0.87) (.515598弧度)

(8) sin10° + cos20° (1.11334)

(9) sin (32.4° − 14.6°) + tg43.7° (1.68698)

(10) arctg (−1.8) + arctg (3.1) (11.1759)

第四章　BASIC语言的基本概念

BASIC语言是一种高级程序设计语言。通过BASIC语言可以使人同机器对话。COMX—PC1的BASIC语言是通过BASIC解释程序来实现其功能的。象其它高级程序设计语言一样，BASIC语言有自己的字符、字、语句（指令），以及自己的语法规则。

§1　BASIC语言的字符

BASIC语言的字符分为以下四类。

一、大写英文字母

A，B，C，D，E，F，G，H，I，J，K，L，M，N，P，O，Q，R，S，T，U，V，W，X，Y，Z。

二、数字

0，1，2，3，4，5，6，7，8，9

三、其它符号

1．运算符号

\+　加号

\-　减号或负号

*　乘号

/　除号

∧　乘方号

（　左括号

）　右括号

=　赋值号

2．关系符号

=　等于号

>　大于号

<　小于号

<>　不等号

>=　大于或等于号

<=　小于或等于号

3．功能标点符号

,　逗号，分界符号

;　分号，打印分界符号

:　冒号，分隔语句符号

.　小数点符号

"　双引号，字符串分界符号

4．专用符号

$　字符串变量符号

␣　空格符号（屏幕上不显示，有时为了书写方便用此符号）

5．特殊符号

#　井号

/　斜杠号

@　花号

■　方块号

这些特殊符号对于不同机型有各种不同的用途，在COMX—

PC1电脑中会用到#、＼和■号。

§2　BASIC语言的字

BASIC字一般也称为BASIC保留字。COMX—PC1电脑的所有 BASIC 保留字列在书后的附录中。这里只列出一些最基本的BASIC字。

LET，PRINT，INPUT，GOTO，IF，THEN，READ，DATA，RESTORE，REM，END，DIM，FOR，TO，STEP，NEXT，GOSUB，RETURN，LIST，NEW，RUN。

§3　BASIC语言的常数、变量、函数和表达式

一、BASIC常数

1．数值常数

在电脑中数值常数（简称常数）有两种形式：整数型和实数（浮点数）型。无论哪种形式的常数均为十进制数。COMX BASIC 中的整数范围是 －2147483647 至 2147483647；实数范围是-0.170141×10^{39}至0.170141×10^{39}。

COMX—PC1电脑常态为实数方式。此时，屏幕上显示的数为6位有效数字。绝对值大于999999或绝对值小于0.1的实数用科学计数法表示，即用10的 n 次幂表示。例如，17824000在实型时表示为.17824E+08。小数点左边的

0不显示，E+08表示10的8次幂。如果实型数超出允许范围，将会出现错误信息。通常，除需要有效位很多的整数运算外，我们都使用实数形式来表示常数。

2．字符串常数

由于电脑需要完成大量的文字处理工作，所以 BASIC 语言也提供了字符串常数（简称字符串）。字符串是除双引号以外的字符的集合。字符串需要用双引号“ ”括起来。例如“COMX”，“BEIJING”等。双引号是字符串分界符，不能做字符串的一部分。

字符串一般用来表示人机对话的语言或注释程序。有时也用做运算的辅助工具。

无论是数值常数还是字符串常数在电脑工作时都是不变的。有时为了方便也称它们为数据。

二、BASIC变量

BASIC语言中设置了BASIC 变量。每一个变量有一个名字。一般称为变量名，为了方便就称为变量了。电脑在工作时，其值是可以改变的。BASIC变量可分以下几类。

1．数值变量

a． 简单变量

简单变量的名字可以用任一个英文字母表示，也可以用任一个英文字母打头后面跟数字0至9中的任一个数表示。例如，A，F，B0，Z9 等都是简单变量名。简单变量名一共有286个。简单变量简称变量。

b、数组变量

数组变量是由一类带下标的变量组成。详细介绍见第八章。

2．字符串变量

a．简单字符串变量

简单字符串变量是由一个简单变量后面跟一个“$”符号构成。例如，A$，C1$ 等都是简单字符串变量。简单字符串变量简称串变量。

b、字符串组变量

字符串组变量由带下标的数组变量加上“$”组成。例如A$（3）、C$（6）等。详细介绍见第八章。

三、BASIC函数

BASIC 语言设立了许多函数，除在上一章介绍的各类函数外，还有随机函数 RND，打印格式函数TAB及一些字符串函数，比如MID $等。COMX BASIC 设立了更多的函数，在以后各章将详细介绍。

四、BASIC表达式

BASIC表达式一般分为算术表达式和关系表达式。

1．算术表达式

由变量、常数、函数、运算符等构成的式子叫算术表达式。例如，代数式 $3x^2+x-\sin x+1$ 写成算术表达式为 3*X∧2+X-SIN（X）+1。半径为r的圆周长写成算术表达式应为2*PI*R。底为b，高为h的三角形面积的算术表达式应为B*H／2。

2．关系表达式

由关系符号把算术表达式联接起来，构成的式子称关系表达式。例如， X+Y=SQR（Z），3*Y+2>=5*SIN（A）都是关系表达式。关系表达式多用于条件语句中，其用法也是相当灵活的。

在 BASIC 表达式中，构成表达式的各个元素不必同时出现。

§ 4　BASIC 程　序

一、行号

每个 BASIC 语句或指令前的数字称为行号。COMX BASIC允许的行号数值为 0 至65535之间的自然数。例如，5，20，98，5076 等均可做为行号。一般来说，在 BASIC 程序中，行号是不能缺少的。

二、BASIC语句

由行号、BASIC 字、变量、算符、表达式（不都是必要的）按一定的语法结构组成的式子叫 BASIC 语句。BASIC语句的功能是使电脑完成某一个特定的动作。一般情况一行只写一个BASIC 语句。例如：

```
10  PRINT  5 * 6
```

是一个有打印输出功能的BASIC语句，执行这个语句后屏幕上会显示出 5 乘 6 的结果。再如

```
90  END
```

也是一个BASIC语句。它的功能是使电脑停止工作。

三、BASIC程序

程序就是指令的有序集合，由若干个BASIC 语句按某种特定的逻辑关系排列的语句集合构成了BASIC程序。

电脑执行BASIC 程序，一般是按行号的顺序由小到大顺序执行（有时也可能跳跃）。由于BASIC解释程序是边

解释边执行BASIC语句，因而BASIC程序运行起来速度不很快。但BASIC程序占用内存少，并且便于修改，易学易用。因此，用BASIC语言编写的程序用途非常广泛。

第五章　最基本的BASIC语句

最基本的BASIC语句包括LET、PRINT和INPUT语句以及NEW、LIST和RUN指令。

§1　语句、指令简介

一、LET语句

LET语句也称赋值语句。LET语句的功能是将表达式的值赋给变量。在处理字符串的时候，赋值语句是将字符串赋给字符串变量。

二、PRINT语句

PRINT语句称为打印语句。PRINT语句的功能是在显示屏上输出变量或表达式的实际值，并能将字符串原样显示。这个语句具有运算和输出双重功能，在遇到表达式时，它先计算后输出。

三、INPUT语句

INPUT 语句称为键盘输入语句。INPUT 语句的功能是等待电脑从键盘上获取信息，然后把从键盘上得到的值或字符串赋给变量或字符串变量。

四、NEW指令

NEW指令是清除指令。它将使电脑内存中的BASIC程序全部消失。具体地说，它清除内存中的用户程序区，但不清除数据区和变量中的值。

五、LIST指令

LIST指令也称列清单指令。它的功能是将BASIC程序的清单显示在屏幕上。根据显示的信息，可以检查程序是否正确。凡是经LIST指令显示出的信息才是电脑内存程序区中的内容。

六、RUN指令

RUN指令也称执行指令，它指挥电脑开始执行BASIC程序。

§2 赋值语句

赋值语句一般有以下几种格式。

1．行号 LET 变量=算术表达式

2．行号 LET 串变量="串常数"

3．行号 LET 串变量=串变量+串变量+……

赋值符号左边只能是变量或串变量。而赋值符号的右边可以是算术表达式、变量、常数或函数。在格式3中，赋值符号右边的加号是联接符号，其功能是将两个串变量联起来。

例 5.1 赋值语句

```
10 LET A=3
20 LET B=5*6
25 LET C=A+B
30 LET A$="ABC"
40 LET B$="DEF"
```

50 LET C$ = A$ + B$

第10行的作用是将3赋给变量A。第20行的作用是将5乘6的结果30赋给变量B。第25行的作用是将A的值3和B的值30相加后的结果33赋给变量C。第30、40行的作用一样，是将字符串ABC和DEF分别赋给串变量A$和B$。第50行的作用是将A$和B$联接起来赋给C$，这时C$是ABCDEF。

目前各类电脑的赋值语句均可省略LET这个保留字。因而，有无LET这个保留字的赋值语句都是一样的。

另外，赋值语句也可以立即执行。其格式同基本格式一样，只是不需要行号。立即执行的赋值语句一般用途不大，只是在调试程序时才用到。

§3 打印语句

打印语句一般有以下几种格式。

一、固定格式输出

PRINT语句固定输出的格式为：

行号PRINT表达式，表达式……

式中逗号“，”的作用是使输出信息按固定的标准格式显示。COMX—PC1电脑显示屏上，有以下两种标准格式。

1．不大于8个字符输出，每一行有5个标准打印位置，相邻的两个位置之间空8位间隔，例如

PRINT 1，2，3，4，5，6

输出显示为：

1　　　2　　　3　　　4　　　5　　　6

2．大于8个字符输出，每行有二个位置，位置间隔为6位。例如：

PRINT　0.875321 E15，－0.123456 E－10，"ABCDEFGHIJKLMN"

输出显示为

.875321 E＋15　　　　－0.123456 E－10

ABCDEFGHIJKLMN

二、紧凑格式输出

PRINT语句紧凑输出的格式为：

行号 PRINT 表达式；表达式……

式中分号"；"的作用是使其前后字符输出为紧凑格式，例如：

PRINT 1；2；3；4；5；6

输出显示为

1 2 3 4 5 6

又如：

PRINT "A＝"；5

输出显示为

A＝5

三、换行输出

PRINT 语句换行输出的格式为：

行号 PRINT 表达式

式中表达式后不带任何符号。例如：

```
10  PRINT  3 * 5
20  PRINT  4 * 6
```

```
30  END
```

该程序执行后的结果为：

```
15
24
```

这就实现了换行输出。

四、字符串输出

PRINT 语句的字符串输出格式为：

行号 PRINT“字符串”

或

行号 PRINT 字符串变量

式中双引号（“ ”）是字符串输出的分界符，它不能做为字符串的一部分。凡是在双引号内的字符都会原样显示出来。例如：

```
PRINT "A+B"
```

输出显示为

```
A+B
```

输出为串变量时，会将串变量所包含的字符串全部显示出来。例如：

```
10  A$ = "ABCD"
20  PRINT  A$
30  END
```

程序执行后的结果为

```
ABCD
```

五、格式函数输出

PRINT语句可以根据格式函数TAB的规定按一定格式输出。TAB函数用于PRINT语句的格式为：

行号 PRINT TAB（表达式）；表达式

式中TAB函数中表达式的值应在 0 至39之间。其功能就是在紧接该值确定的位置后开始打印输出。例如：

PRINT TAB（17）；12

将在第18列开始打印显示12这个数。

TAB函数中表达式的值在COMX—PC1电脑上只能取 0 至39中的数。如果所取的值为小数，函数会自动取整。TAB函数可在PRINT语句中连续使用，但一定注意要用"；"做为分界符。例如：

PRINT TAB（5）；"ABC"；TAB（11）；"DEF"

输出显示为

```
          ABC      DEF
```

六、混合输出及光标定位

上述几种输出都可以混合联用。

例 5.2 输出格式混合联用

```
10  PRINT "A", "B", "C"
20  A = 4
30  B = 5
40  C = 6
50  PRINT  A, B, C
60  PRINT
70  PRINT "A ="; A, "B ="; B, "C ="; C
80  END
```

程序运行后的结果应为：

```
A                  B                  C
```

4　　　　　　　5　　　　　　　6

A = 4　　　　　B = 5　　　　　C = 6

程序中第60行有一个空PRINT语句，它的功能是输出一个空行，并且将光标移至下一行的左边起始处，在图形和表格输出时经常用到空PRINT语句。

COMX—PC1电脑还专门设置了控制光标指令：

CPOS（x，y）

其中x，y分别为行、列参数，可以是表达式、变量、常数或函数。x的取值范围是0至23，y的取值范围是0至39。CPOS指令可使光标处于屏幕的任一位置，这样，就使输出打印随心所欲。

如果在CPOS指令后不接任何语句，那么光标只能移行。一般情况下，CPOS指令后都紧跟有PRINT语句或CLS指令。例如：打入

CPOS（11，17）：PRINT "HELLO"

输出的HELLO将显示在屏幕中央。即在第12行第18列处开始打印HELLO。再打入

CPOS（11，17）：CLS

执行后HELLO就会被清除。

CLS是COMX—PC1电脑具有的清屏指令。它从光标处开始清屏，直至屏幕的最下一行。最常用的清屏方法是：

CPOS（0，0）：CLS

它表示将全屏幕的内容完全予以清除。

上边的几个语句中，用到了冒号"："。冒号是语句或指令分隔符号，实际上，上边的几个例子中都是两个指令或

语句。中间由“：”分开。在BASIC程序中，一个行号可写几个语句，注意须用冒号隔开。

例 5.3 写一个BASIC程序，计算半径分别为5cm、7.5cm的圆的面积和周长。

利用公式　圆面积 $=\pi R^2$

圆周长 $=2\pi R$

我们先写出半径为5cm的圆面积和周长的程序。

```
10  R = 5
20  A = PI*R*R
30  C = 2*PI*R
40  PRINT  A
50  PRINT  C
60  END
```

在程序输入前应先打入NEW指令。注意不要忘记按CR键。

现在按以下步骤输入程序。即

```
NEW                 〔CR〕
10  R = 5           〔CR〕
20  A = PI*R*R      〔CR〕
30  C = 2*PI*R      〔CR〕
40  PRINT  A        〔CR〕
50  PRINT  C        〔CR〕
60  END             〔CR〕
```

如果想要检查输入的程序有没有错误，请打入

```
LIST        〔CR〕
```

此时，如果没错，电脑屏幕上会出现你输入的程序：

```
10  R = 5
```

```
20  A=PI*R*R
30  C=2*PI*R
40  PRINT  A
50  PRINT  C
60  END
```

如果你的程序输入有错，请改正它。哪一行有错就改哪一行，即重新打入该行，特别注意行号要正确，新打的一行与错行的行号要一致。最后再用LIST指令检查，直至完全正确为止。

运行这个程序只要打

RUN 〔CR〕

屏幕上就会出现：

```
78.5398
31.4159
```

说明半径为5 cm的圆面积是78.5398cm²，其周长是31.4159 cm。

在输入每一行程序时，都要按CR键。这是因为只有按CR键后，输入的信息才能进入电脑内存中去。如果在一行语句后不按CR键，则会出现类似

```
10  R=5    20  A=PI*R*R    30……
```

这样的错误语句。电脑是绝不会接受这样的语句的。初学者常犯的毛病就是忘记按CR键。特别是打完程序的最后一行后，急于运行，匆忙打入RUN，比如最后一行是

```
60  END
```

打成了

```
60  END RUN
```

这完就全错了。还需要重新打入这一行。结果是欲速则不达。

现在来计算半径为7.5cm的圆面积和周长。不必重新输入程序，只需将

```
10  R= 5
```

改为

```
10  R=7.5                [CR]
```

即可。然后检查程序是否改好。

打入

```
LIST                     [CR]
```

屏幕上应显示：

```
10  R=7.5
20  A=PI*R*R
30  C= 2*PI*R
40  PRINT  A
50  PRINT  C
60  END
```

显然，第10行已经改好了。电脑的特点之一就是后入为主。此时变量R的值是多少呢？打

```
PRINT  R                     [CR]
```

结果显示为

```
5
```

怎么是5，不是已经改过了吗？原因是这样的：虽然新的语句已经输入，但电脑只是将该句存在内存中，并没有执行它，只有重新执行RUN指令，才能使10语句起作用。在执行RUN之前，R的值仍然是5。现在打

```
RUN                    〔CR〕
```

屏幕上出现：

```
176.715
47.1239
```

说明半径为7.5cm的圆面积为176.715cm²，周长为47.1239cm。

上面程序的输出结果不是很直观。为了改善它，我们加入下面几行程序。

```
35  PRINT   "R="        〔CR〕
36  PRINT  R            〔CR〕
38  PRINT  "A="         〔CR〕
45  PRINT  "C="         〔CR〕
```

现在打

```
LIST                    〔CR〕
```

屏幕上显示出：

```
10  R=7.5
20  A=PI*R*R
30  C= 2*PI*R
35  PRINT "R="
36  PRINT  R
38  PRINT "A="
40  PRINT  A
45  PRINT "C="
50  PRINT  C
60  END
```

现在打

```
RUN                         〔CR〕
```

结果为：

```
R =
7.5
A =
176.715
C =
47.1239
```

这个输出形式也不理想，现将40行改为：

```
40  PRINT "R = "; R, "A = "; A , "C = "; C
```

然后将35，36，38，45，50行删去。打

```
35                           〔CR〕
36                           〔CR〕
38                           〔CR〕
45                           〔CR〕
50                           〔CR〕
```

然后用LIST指令检查是否改好，最后打

```
RUN                         〔CR〕
```

结果为：

```
R = 7.5          A = 176.715          C = 47.1239
```

这样的输出结果就比较醒目了。

§4 键盘输入语句

键盘输入语句的格式有下列三种：

1. 行号 INPUT变量｛，变量…｝

2．行号　INPUT串变量

3．行号　INPUT“字符串”变量

在执行INPUT语句时，屏幕上会显示出一个“？”，这是电脑等待输入数据的信号，每次打入数据后，应按CR键，这样才能完成赋值工作。如果数据是字符串，不需要用双引号。

例 5.4　在直角三角形中。已知两直角边长，求第三边的长度。

已知a＝3，b＝4；a＝7，b＝8。利用公式

$$C=\sqrt{a^2+b^2}$$

编写这个程序的方法很多，下面介绍三种供参考。

程序 1

```
10  A= 3
20  B= 4
30  C=SQR (A*A+B*B)
40  PRINT "A="; A, "B="; B, "C="; C
50  END
```

执行后的结果为：

```
A= 3        B= 4        C= 5
```

要计算第二组数据需将第10行、第20行改为

```
10  A= 7
20  B= 8
```

重新打

```
RUN
```

结果为：

```
A= 7        B= 8        C=10.6301
```

如果我们再算几组数，每次都改变程序是很麻烦的。INPUT语句提供了方便的办法。

程序 2

```
10  INPUT  A, B
20  C=SQR (A*A+B*B)
30  PRINT "A="; A, "B="; B, "C="; C
40  END
```

打

```
RUN
```

屏幕上出现

```
?
```

打入

```
7, 8                    [CR]
```

结果为：

```
A= 7        B= 8        C=10.6301
```

该程序还可以有另一种编写方法。

程序 3

```
10  INPUT "A=" A
20  INPUT "B=" B
30  C=SQR (A*A+B*B)
40  PRINT "C="; C
50  END
```

运行后为：

```
A=? 14                  [CR]
B=? 20                  [CR]
C=24.4131
```

§ 5　几个值得注意的问题

一、任何一个指令或语句想要让电脑接受，必须要按CR键。

二、程序输入执行步骤：

1．先打NEW；

2．输入程序，不要忘记打行号，语句输入先后不限；

3．检查程序用LIST；

4．有错及时改正；

5．执行程序用RUN。

三、改错要点

1．在按CR键以前发现错误可以用DEL键和CTRL—R键来改正错误。假如在下面语句中误打了一个字母（R→A），

10　PAINT　37◆

此时还没有按CR键，可以按DEL键若干次，使光标移至紧接P的右面，即

10　P◆

打入R，显示为

10　PR

按CTRL—R，此时屏幕上会显示出

10　PRINT　37◆

说明已经改好了，可以按CR键了。

2．在按CR键之后立即发现了错误，可以按CTRL—R键，然后再按上面的方法改正错误。例如下面的错句为

（冒号误打为分号）

300　CPOS（0，0）；CLS

此时刚按过CR键，只要再按CTRL—R键就会显示出

300　CP S（0，0）；CLS♦

用DEL键四次，重打：CLS，然后按CR键，这个语句输入就正确了，即

300　CPOS（0，0）：CLS

3．如果在LIST指令之后，发现了错误，最简单的改错方法是重新输入该行。比如，程序

```
10  A = 3
20  B = 5
30  PRINT  A + B
40  PRINT  AB
50  END
```

第40行不对，只要打入

```
40  PRINT  A * B
```

即可。

四、行号间隔

一般来说，行号间隔视个人习惯而定。如果程序很成熟，可以连续使用行号，如1，2……。但通常为了便于修改程序或插入新的语句，在行号选择上应留有一定的余地。

有时程序修改后，行号的间隔就会不一致了，读起来不太方便。COMX—PC1电脑提供了重新整理行号的指令。其格式有下列两种：

1．RENUMBER

2．RENUMBER　n

格式 1 表示整理后的行号以10为间隔。格式 2 中n为一个自然数，这种格式表示整理后的行号以n为间隔。例如，程序

```
1  A = 5
2  C = 6
9  PRINT  A + C
13 END
```

打

RENUMBER　5后，

程序行号已经按 5 为间隔重新整理了。打入

LIST

显示为：

```
5  A = 5
10 C = 6
15 PRINT  A + C
20 END
```

§6　综合例题

例 5.5　变量赋值

```
10 A = 10
20 B = 7
30 C = A + B - 3
40 D = A - B + 4
50 PRINT"A = ";A,"B = ";B,"C = ";C,"D = ";D
60 A = 6
```

```
70  D=B*A
80  C=D-B
90  PRINT "A=";A, "B=";B, "C=";C,
      "D=";D
100 END
RUN
A=10        B= 7        C=14        D= 7
A= 6        B= 7        C=35        D=42
```

例 5.6 两数互换

```
10  I= 7
20  J= 9
30  PRINT "I=";I, "J=";J
40  K=I
50  I=J
60  J=K
70  PRINT "I=";I, "J=";J
80  END
RUN
I= 7        J= 9
I= 9        J= 7
```

两数互换必须通过一个中间变量，这个例子中中间变量是K。

例 5.7 求5个数的平均值

```
10  INPUT A, B, C, D, E
20  M=(A+B+C+D+E) 15
30  PRINT "M=";M
```

```
40  END
RUN
? 1, 2, 3, 4, 5
M = 3
```

例 5.8 打 EP HELLO 和 HOW ARE YOU? 编写这个程序可以用多种办法，通过这个练习，我们可以领会几个基本语句的特点。

程序 1

```
10  PRINT "HELLO"
20  PRINT "HOW  ARE  YOU? "
30  END
RUN
HELLO
HOW  ARE  YOU?
```

程序 2

```
10  A$ = "HELLO"
20  B$ = "HOW  HOE  YOU? "
30  PRINT  A$
40  PRINT  B$
50  END
RUN
HELLO
HOW  ARE  YOU?
```

程序 3

```
10  INPUT  A$
20  INPUT  B$
```

```
30  PRINT  A$
40  PRINT  B$
50  END
RUN
? HELLO
? HOW  ARE  YOU?
HELLO
HOW  ARE  YOU?
```

输入字符串不需用双引号。

例 5.9 与电脑对话

```
10  PRINT "I  AM  A  COMPUTER".
20  INPUT "WHAT  IS  YOUR  NAME" N$
30  PRINT
40  PRINT "HELLO, "; N$
50  PRINT "NOW I KNOW YOUR NAME! "
60  PRINT "I LIKE"; N$; "AS A NAME. "
70  END
RUN
I AM A COMPUTER.
WHAT IS YOUR NAME? LI WING
HELLO, LI MING
NOW I KNOW YOUR NAME!
I LIKE LI UING AS A NAME.
```

§7 COMX—PC 1 电脑的屏幕和字符颜色

一、屏幕颜色

COMX-PC1电脑屏幕具有八种颜色，由指令SCREEN决定。其格式为

行号 SCREEN（表达式）

表达式的值为1至8之间的任一整数。SCREEN 可以立即执行。例如

SCREEN（3）

屏幕变成蓝色

SCREEN（5）

屏幕变成红色。表5.1给出了参数与屏幕颜色及亮度的对应关系。

表 5.1　　屏幕颜色

参数	屏幕颜色	亮度（%）
1	黑色	0
2	绿色	59
3	蓝色	11
4	蓝绿色	70
5	红色	30
6	黄色	89
7	洋红色	41
8	白色	100

二、字符颜色

字符颜色也可以改变，COMX—PC1电脑的字符分成输入字符和输出字符。两类字符颜色也不相同。改变字符颜色用COLOR指令完成，其格式为

行号　COLOR（表达式）

表达式的值可取1至12之间的任一整数，例如

COLOR（2）

键盘输入为黄色，电脑响应为红色。

COLOR（8）

键盘输入为白色，电脑响应为黄色。表5.2列出了参数、键盘输入颜色及电脑响应颜色之间的关系。

表 5.2　　　字　符　颜　色

参　数	键盘输入		电脑响应	
	颜　色	亮　度（%）	颜　色	亮　度（%）
1	绿色	59	黑色	0
2	黄色	89	红色	30
3	蓝绿色	70	蓝色	11
4	白色	100	洋红色	41
5	蓝色	11	黑色	0
6	洋红色	41	红色	30
7	蓝绿色	70	绿色	59
8	白色	100	黄色	89
9	红色	30	黑色	0
10	洋红色	41	蓝色	11
11	黄色	89	绿色	59
12	白色	100	蓝绿色	70

三、色调调节

色调调节用指令CTONE来实现，该指令可以变换屏幕和字符的色调。其格式为

行号　CTONE（表达式）

表式式的值为1使字符颜色变深，值为0消除取值为1的效果。

以上几个指令一般应用于教学程序和游戏程序中，可使图形丰富多彩，增加立体感。如果使用的显示器不是彩色的，显示差别只表现在深浅上。

例　5.10　编一程序显示屏幕的全部颜色

```
10  FOR  I=8  TO  1  STEP  —1
20  SCREEN (I)
30  WAIT (100)
40  NCXT  I
50  END
```

程序中10、40行为循环语句，30行为等赶语句。运行这个程序，会看到屏幕上颜色的交换。

§8　练　习　题

一、编一个简单程序，求一元二次方程的实根（设方程存在实根）。参数自己定。

二、编一个简单程序，当A＝5，B＝4，C＝9时求Y＝（AB－C）／（4C－AB）＋AC／B的值。

三、已知任意三角形的二边长a、b和一夹角x，求第三边，公式$C=\sqrt{a^2+b^2-2ab\cos x}$。其中a＝14，

$b=40$，$x=25°$。

四、用INPUT语句写出一个将厘米变换为英寸的程序（换算关系1英寸=2.54厘米）并转换下列数据：2.54cm，78.8cm，172.8cm，180cm。

五、根据运动学公式$S=V_0t+at^2/2$，编写一个求距离S的程序并计算S。其中a=1米／秒^2，V_0=5米／秒，t分别是1秒，1.5秒，2秒，2.5秒，3秒，3.5秒，4秒。

第六章　条件判断

§1　条件分支的引入

在第五章§4例5.4的程序3中，每运行一次程序会得到一次结果。如果想多算几道题，就要打多次RNU命令，这当然是很麻烦的。BASIC语言条件分支语句可以给多次运行的程序提供方便。

例6.1　修改例5.4的程序

```
10  INPUT "A=" A
20  INPUT "B=" B
30  C=SQR (A*A+B*B)
40  PRINT "C="; C
50  IF A<>0 THEN GOTO 10
60  END
```

下面是执行这个程序的典型例子

```
RUN
A=? 3
B=? 4
C=5
A=? 6
B=? 8
C=10
A=? 0
B=? 0
```

```
C=0
READT
:♦
```

下面我们来看一下这个程序是如何执行的。COMX—PC1电脑允许使用TRACE命令跟踪程序，打入

```
1  TRACE  1
```

然后打

```
RUN              表示开始运行
TR〔10〕         表示执行第10行
A=? 3            输入3
TR〔20〕         表示执行第20行
B=? 4            输入4
TR〔30〕         表示执行第30行
TR〔40〕         表示执行第40行
C=5              输出结果 5
TR〔50〕         表示执行第50行
TR〔10〕         表示执行第10行
A=? 6            ……
TR〔20〕
B=? 8
TR〔30〕
TR〔40〕
C=10
TR〔50〕
TR〔10〕
A=? 0
```

TR〔20〕

B=? 0

TR〔30〕

TR〔40〕

C= 0

TR〔50〕　　　　　　因为A=0，不再转向。

TR〔60〕　　　　　　表示执行第60行

显然，电脑执行每一行程序都可以通过TRACE命令来实现跟踪。一般来说，在编写完一个程序后，自己总应该“走一走”程序，这样在上机调试前就会将明显的错误清除，以便提高效率。在上机调试中可以使用 TRACE 命令来跟踪程序。

TRACE 命令的格式为：

行号 TRACE 表达式

当表达式大于零时建立跟踪，当表达式等于零时取消跟踪。一般来说，TRACE命令的行号都是所要调试程序的最小行号（0除外）。

§2 条件语句

条件语句就是根据所给出的条件是否满足，而决定应该执行哪条指令。

一、条件语句的格式

条件语句在COMX BASIC中的标准格式是：

行号　IF　表达式1关系符　表达式2

THEN　可执行的BASIC语句

表达式可以是常数、变量、函数。可执行的 BASIC 语句包

括LET、PRINT、GOTO等语句。

字符串也可以进行比较，不过关系符只能使用“=”和“<>”两种。

二、条件语句浅释

条件语句的使用象到了叉路口选择道路一样，根据条件的满足与否决定程序的走向。示意图见图6.1。条件语句的功能实际上是：满足条件就执行THEN后的BASIC语句；不满足条件就执行条件语句所在行后面的最小行号的那条语句。例如：

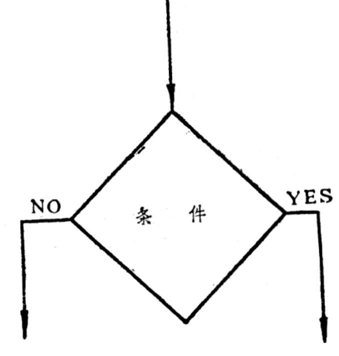


图 6.1 条件语句示意图

```
50  IF A<>0 THEN GOTO  10
60  END
```

如果A不等于0就去执行第10行。如果A等于0就去执行第60行停机。

以下几个语句都是典型的条件语句。

```
100  IF  X+Y>3.8  THEN  GOTO  50
170  IF  A$ = "A"   THEN  PRINT  A$
```

```
450 IF C$<>"ABC" THEN C$="ABC"
```

在COMX BASIC中THEN这个保留字可以省略。例如：

```
50 IF A<>0 THEN GOTO 10
```

与

```
50 IF A<>0 GOTO 10
```

是等价的。但要切记，诸如

```
50 IF A<>0 THFN 10
```

这样的语句是不能被COMX—PC1电脑接受的，因为10不是一个可执行的BASIC语句，这里的GOTO是不能省略的。

COMX—PC1电脑允许在条件语句后联接其他语句，但要用冒号隔开。例如

```
40 IF A=C THEN E=C：PRINT C：GOTO 70
50 A=B
60 PRINT A
70 END
```

程序40行表示：如果条件A>C满足，将执行赋值语句A=C，然后执行PRINT C，接着转向70行；如果条件不满足，将顺序执行50、60、70行。

三、COMX—PC1电脑的逻辑算符

逻辑算符AND、OR、NOT、XOR在COMX—PC1电脑中是不能随便用的。

AND和OR的功能可以采取复合条件语句来完成。

AND的功能可采用串联IF来实现，例如：

```
30 IF A<5 IF A>0 THEN PRINT A
```

表示如果条件A＜5和A＞0同时满足则执行PRINT A 语句；有一个条件不满足都不去执行PRINT A语句。

OR的功能可用并列IF来实现，例如：编写一个“当C等于7或C等于13时就打印1，否则就打印0”的程序。程序如下：

```
10  INPUT C
20  IF  C=7   THEN  PRINT  1:END
30  IF  C=13  THEN  PRINT  1:END
40  PRINT  0
50  END
```

这个程序还可写成

```
10  INPUT  C
20  IF  C=7   THEN  GOTO  60
30  IF  C=13  THEN  GOTO  60
40  PRINT  0
50  END
60  PRINT  1
70  END
```

在程序中使用多个END语句是允许的。

例 6.2 编写一个程序，求整数集合中从零至N前N+1个整数之和。N由键盘输入。下面介绍两种编法。

程序 1

```
10  INPUT "N=" N
20  I=0
30  S=0
40  S=S+I
```

```
50  I=I+1
60  IF  I<=N  THEN  GOTO  40
70  PRINT "SUM=";S
80  END
```

计算N=10的结果：

```
RUN
N=? 10
SUM=55
```

程序2

```
10  INPUT "N=" N
20  I=0
30  S=0
40  I=I+1
50  S=S+I
60  IF  I<N  THEN  GOTO  40
70  PRINT "SUM=";S
80  END
```

计算N=100的结果：

```
RUN
N=? 100
SUM=1050
```

这两个程序原理是一样的。只是第40行和50行次序对调了。第60行的条件略有不同。语句I=I+1是一个加1语句，即每执行一次这个语句，原有变量I的值就要增加1。语句S=S+I起累加器的作用，它相当一个存储库，每次存入的数值都要与原有数值相加。下面简单分析一下这两个程序。

程序1在执行后首先输入N值（10）。然后执行I＝0和S＝0，这是将两个变量清零。40行执行后S仍为0。50行执行后I为1。60行的条件满足转回执行40行。执行40行后S为1。再执行50行I值为2。60行条件语句执行后仍回40行。此时执行40行后S值是1＋2，即为3。……直至I为11时，条件不满足，则输出结果。

程序2的运行同程序1大致相同，只是因为它先执行加1语句，所以待I值达到N值后就可以输出了。因而条件语句中用了小于符号。另外，程序2比程序1少执行一次40、50、60语句。

§3 GOTO语句

GOTO语句是一个无条件转移的可执行语句。GOTO语句的格式为：

行号　GOTO　表达式

通常表达式的值一定是程序中有的行号。

GOTO语句的作用就是不论什么情况都要按指定的行号去执行转移的命令。因而也称为强迫转移。在GOTO语句的表达式中，如果不是整数，电脑会自动取整，然后去执行相应的语句。

在程序中一定要保证转向的行号存在，否则会出现错误信息。

为了保证整个程序的完美，应尽量少用GOTO语句。如果不得不用，则尽量转向行号较小的语句。这样可以加快程序执行的速度。

另外，COMX—PC 1 电脑提供了一个加快执行GOTO语句的命令：

RUN+

此命令与RUN不同，它首先检索你程序中的转移语句，变解释转移方式为绝对地址转移方式，然后再开始执行程序。这样可以加快程序运行的速度。

§4 REM语句和RND函数

当程序逐渐变得复杂时，很难看出程序中某些语句是在处理什么，这就需要注释。为了使程序更加丰富多采，就需要有一个产生各种数的函数。BASIC 语言给你提供了这样的语句和函数。

一、注释语句

REM 语句就是注释语句。凡是要加以说明的文字都可以借助REM语句来实现。其格式为：

行号 REM 注释字符

REM语句在程序中只起注解作用，BASIC系统不执行这个语句，一遇到它就会自动跳过去。

例如下面的语句就是一个注释语句

10 REM 20—30 OUTPUT ANSWER

注释为20至30为输出答案。

REM语句中的注释字符没有任何限制。但编写程序时，应力求注释简明，否则会占用过多的内存空间。

二、随机函数

随机函数也称随机数，一般用RND表示。它可以产生一

系列随机数。

RND有许多用途，常用于模拟、游戏以及数学程序中。如果我们向空中抛100次硬币，当它落到地面上时，有多少次是正面向上？又有多少次是反面向上呢？这个问题可以利用RND来模拟。又如，在教学程序中，让电脑自动地提出各种不同的算术题也可用RND来实现。COMX—PC1电脑的RND基本分为两种。

1．浮点型

RND

它产生0至1之间的小数，可以表示为

$$0 < RND < 1$$

它不同于其它电脑的RND。

例 6.3 产生10个随机数

```
10  S=0
20  S=S+1
30  A=RND
40  PRINT  A
50  IF  S<10  THEN  GOTO  20
60  END
```

执行后一种可能的结果是：

.519922 .87519 .571882 .554516 .180149

.090253 .920173 .708761 .171021 .357216

再次执行就不一定是这个结果。也许第一次执行也不是这个结果，这是正常的。它只是产生10个随机数而已，具体是什么数是不一定的，但肯定是0至1之间的浮点数。

2．整型

RND（表达式）

表达式的值为一个整数，如果不是整数，会自动取整。

整型RND产生一个大于0或等于0且小于表达式值的随机整数，可以表示为

0⩽RND（表达式）＜|表达式|

如果应用整型RND，必须同时考虑采用整型变量，这将在例题中介绍。

§5 综合例题

例 6.4 统计选票

编写一个统计选票的程序。候选人有三个，参加投票人数不限，求出每个候选人各自所得的票数。候选人的姓名由键盘输入给字符串变量A$、B$、C$。字符串A、B、C代表选票的性质（即投哪个人的票）。统计的计数变量由A、B、C表示。字符串R为重复标志，字符串E为输出结果标志。程序如下：

```
10  A=0：B=0：C=0
20  INPUT  "NAME-1" A$
30  INPUT  "NAME-2" B$
40  INPUT  "NAME-3" C$
50  INPUT  "INPUT  A/B/C/E" X$
60  IF  X$="A" THEN  A=A+1：GOTO 50
70  IF  X$="B" THEN B=B+1：GOTO 50
80  IF  X$="C" THEN C=C+1：GOTO 50
90  PRINT "INPUT ERROR OR END(R/E)";
```

```
100 INPUT  Y$
110 IF  Y$ = "R" THEN  GOTO  50
120 IF  Y$ < > "E"  THEN  GOTO  90
130 PRINT  A$; TAB (20) ; A
140 PRINT  B$; TAB (20) ; B
150 PRINT  C$; TAB (20) ; C
160 END
```

程序中的X$为接受输入选票字符的串变量。第 110 行为控制返回50行继续统计的语句。而120行则为输入E而达到最后输出结果而设计的判断。同时防止 100 行输入时发生错误。第130行至150行为输出统计结果。

请读者自己运行这个程序。

例 6.5 算术训练程序

编一程序，让电脑提出一道算术题，由读者回答，答案正确电脑再出新题，否则读者重做原题。读者可以事先确定共出多少题，题目分为加、减、乘、除四种单项计算，不确定出哪一种，并且加、减法均为一位或二位整数计算，乘、除法需使乘（除）数为一位整数，被乘（除）数为二位整数，而且被除数是可以被整除的。程序如下：

```
5   DEFINT  Z
10  A$ = "CORRECT"
20  B$ = "WRONG, TRY  AGAIN"
30  INPUT "PROBLEM  NUMBERS" N
40  I = 0
50  I = I + 1
60  IF  I>N  THEN  END
```

```
70  X=RND (4) + 1
80  GOTO  100*X
100 REM  ADDITION
110 A=RND (100) : B=RND (100)
120 PRINT  A; "+"; B; "=";
130 INPUT  C
140 IF C=A+B THEN PRINT A$ : GOTO 50
150 PRINT  B$
160 GOTO  120
200 REM  SUBTRACTION
210 A=RND (100) : B=RND (100)
220 IF  A>B  THEN  GOTO  240
230 Y=A : A=B : B=Y
240 PRINT  A; "-"; B; "=";
250 INPUT C
260 IF C=A-B THEN PRINT A$ : GOTO 50
270 PRINT  B$
280 GOTO  240
300 REM  MULTYPLICATION
310 A=RND (90) +10 : B=RND (10)
320 PRINT  A; "*"; B; "=";
330 INPUT  C
340 IF C=A*B THEN PRINT A$ : GOTO 50
350 PRINT  B$
360 GOTO  320
400 REM  DIVISION
```

```
410 Y=RND（9）+1：B=RND（9）+1
420 A=Y*B
430 PRINT  A；"/"；B；"="；
440 INPUT  C
450 IF  C=Y  THEN  PRINT  A$：GOTO 50
460 PRINT  B$
470 GOTO  430
```

该程序解释如下。

第5行定义变量A至Z均为整型变量。为后面使用方便，第10、20行给A$，B$赋字符串。70行使X取值为1，2，3，4不定。80行根据X的值转向不同种的计算。比如X为1，转向100行（加法运算）；X为2，则转向200行（减法运算）等等。220、230行保证A大于B。420行保证被除数可以整除。

这个程序用的随机数均为整型数。如果采用浮点型需做如下改动：

```
70  X=INT（4*RND）+1
110 A=INT（100*RND）：B=INT（100*RND）
210 A=INT（100*RND）：B=INT（100*RND）
310 A=INT（90*RND）+10：B=INT（10*
RND）
410 Y=INT（9*RND）+1：B=INT（9*
RND）+1
```

此时第5行可以省略。

请读者自己试运行这个程序。

例 6.6 打印"九九乘法表"。要求个位、十位各自对

齐。输出结果应为：

```
1
2   4
3   6   9
4   8  12  16
5  10  15  20  25
6  12  18  24  30  36
7  14  21  28  35  42  49
8  16  24  32  40  48  56  64
9  18  27  36  45  54  63  72  81
```

程序如下：

```
10  A = 1
20  B = 1
30  C = A * B
40  IF  C<10  THEN  PRINT “  ”;
50  PRINT  C; “  ”;
60  B = B + 1
70  IF  B<=A  THEN  GOTO  30
80  PRINT
90  A = A + 1
100 IF  A<= 9  THEN  GOTO  20
110 END
```

程序中第40行的作用是当C值小于10时，在十位的位置上打一个空格，以保证个位对齐。第50行是输出乘积结果，为保证在同一行输出，末尾有一个分号。第70行是当A等于B时控制转行。第80行作用是取消50行末尾的分号，使下面的输

出另起行。第100行是控制变量A的值，实现预定的输出结果。

请读者试运行这个程序。

§6 COMX—PC 1 电脑的编辑功能

在输入程序（尤其是较长的程序）时，不可避免地要出现各类错误。往往需要重新输入某些语句，既麻烦又浪费时间。为了更快更好地输入程序，COMX—PC 1 电脑提供了EDIT（编辑）命令。EDIT命令的格式为

EDIT X

其中X是所要编辑语句的行号。完成上述命令后就进入编辑状态。

编辑完成后打CTRL—S键，就又重新回到BASIC状态之中了。

下面结合实例说明插入、删除、修改的功能。

一、插入I（Insert）

某程序有一错误语句为：

```
10 IF A=4 THEN 100
```

这个语句对于COMX—PC 1 电脑来说是缺少GOTO这个保留字的。我们来插入这个字。首先必须进入编辑状态，打

EDIT 10

按CR键后屏幕上出现：

```
10 IF A=4 THEN 100
♦
```

此时光标在行号下方，BASIC 提示符消失。按空格棒将光

标移到THEN之后下方。

```
10  IF  A=4  THEN  100
                   ♦
```

接着打I，这时就可以插入新字符了。再打GOTO即：

```
10  IF  A=4  THEN  100
                   ♦GOTO
```

然后按CTRL—S键，屏幕上显示出：

```
10  IF  A=4  THEN  GOTO  100
♦
```

这时再按一次CTRL—S键就又回到BASIC状态了。

```
10  IF  A=4  THEN  GOTO  100
READY
:♦
```

可以利用LIST命令检查一下修改后的语句。如果无误就表明插入编辑完成了。

二、删除 D (Delete)

某程序中有如下错误语句

```
10  PRINT  A, B, :C
```

显然多了一个“:”号，需要删除，打入

```
EDIT  10
```

按CR键后屏幕上显示为

```
10  PRINT  A, B, :C
♦
```

移动光标到B后面的逗号下方，即

```
10  PRINT  A, B, :C
                ♦
```

打入D，然后按空格棒一次，再按CTRL—S键，屏幕上显示为

10 PRINT A，B，C

♦

再按CTRL—S键，就返回到BASIC状态了。

READY

：♦

现在打LIST，若输出显示为

10 PRINT A，B，C

则说明删除工作巳经完成。

三、修改 C (Change)

某程序中的一个语句错打成

100 IF C>I00 THEN PRINT C

这个错误是将数字1错打成字母I了。打入

EDIT 100

按CR键后进入编辑状态

100 IF C>I00 THEN PRINT C

♦

移动光标至>号下面。

100 IF C>I00 THEN PRINT C

♦

打入C，再打数字1，接着按CTRL—S键两遍，就可以回到BASTC状态了。打

LIST 100

屏幕上显示出：

100 IF C>100 THEN PRINT C

至此修改过程结束。

四、混合联用

在混合联用编辑状态下，我们可以完成插入、删除、修改中的任一项或它们的组合，但可以不必返回BASIC状态。

例如有一语句 IF A=50 THEN B=A+1：GOTO 10错打为：

```
40  IF A3= 5  THEN B=A+1；GOTO 10
```

现在先要删除A3变量的3字，然后在5字之后加入一个0，最后改正错误，将"；"号改为"："号。先打入

```
EDIT  40
40  IF A3= 5  THEN B=A+1；GOTO 10
```

将光标移到A3的A字下方，打入D，按空格棒一次，再按CTRL—S键。

```
40  IF A= 5  THEN B=A+1；GOTO 10
 ♦
```

将光标移到5的下边，打入I，再打一个0，然后按 CTRL—S键。

```
40  IF A=50 THEN B=A+1；  GOTO 10
 ♦
```

将光标移到分号前面字符的下方，即

```
40  IF A=50 THEN B=A+1；  GOTO 10
                      ♦
```

打入C，然后打一个冒号"："，再按CTRL—S键。

```
40  IF A=50 THEN B=A+1：  GOTO 10
 ♦
```

全部已改完，打CTRL—S键，则有：

```
READY
:♦
```

再打

```
LIST 40
```

屏幕上显示为：

```
40 IF A=50 THEN B=A+1:GOTO 10
```

完成了混合联用编辑工作。

除了利用 EDIT 命令外，还可以利用 DEL 键，CTRL—R键和CTRL—C键进行编辑工作。

当一个语句打完后，如果认为不需要它（此时未按 CR 键），可按CTRL—C键，则该语句不进入电脑内存。

当输入的两个语句稍有不同时，使用CTRL—R 键可以很方便地在第一个语句的基础上输入第二个语句。例如

```
10 IF A>1 THEN GOTO 50
20 IF B>1 THEN GOTO 50
```

这两个语句只有变量的差别，可以先输入

```
10 IF A>1 THEN GOTO 50
```

按CR键后打

```
20 IF B
```

接着按CTRL—R键，会出现

```
20 IF B>1 THEN GOTO 50 ♦
```

按CR键后，第20行语句就输入完成了。

总之，灵活运用COMX—PC 1 电脑的编辑功能会大大提高调试程序的能力。

§7 练 习 题

一、编一程序，对于输入互不相等的三个数，按从大到小的顺序打印出来。

二、编写用 RND 函数模拟投硬币的程序。大于 0.5 的为正面，打印出H，并记次数；小于 0.5 的为反面，打印出F，并记次数。“投币”次数自己确定。

三、编一个程序，打印出 100 之内的素数，并求出这些素数之和。

第七章 处理数据

在程序设计中，常常需要处理大量的数据，COMX BASIC提供了处理数据的语句。

§1 READ、DATA 语句

READ、DATA 语句是用来处理大量数据的。尤其适用于统计、数据分析等题目。

一、READ 语句

READ语句可完成数据读入及赋值工作。READ语句的格式为：

行号 READ 变量 {，变量……} 这里的变量包括所有的BASIC变量。

二、DATA 语句

DATA 语句是数据语句。它是将数据存在内存的程序区中。DATA 语句的格式为：

行号 DATA 常数 {，常数……}

这里的常数包括字符串常数，但使用时需要用双引号。

使用 DATA 语句时，无须考虑行号大小，也就是无须考虑它在程序中的位置。不过，按习惯常进行如下处理：一是将DATA语句放在READ语句之后；二是将DATA 语句放在程序的最后。

三、READ、DATA语句的使用

READ语句与 DATA 语句，在程序中至少要同时出现一次。由于这两个语句各自都能使用多个变量或数据，这就会造成两个语句不成对出现的情况。一般来说，DATA 语句出现得多一些。

在使用READ、DATA语句时，一定要注意数据类型同变量类型相一致。例如

```
10   READ  A, B, C$
100  DATA  3.4, 7, "HELLO"
200  PRINT  A, B, C$
210  END
```

执行后的结果为

```
3.4          7          HELLO
```

数据区中的数据是以前至后存放的，即行号小的DATA语句中的数据存在前面，行号大的DATA语句中的数据存在后面。READ 语句执行时是按照数据指针的位置读取数据的。每次读完一个数据，数据指针会自动移到下一个数据位置上。

例 7.1 对应读数

```
10 READ  A, B, C
20 READ  A$, B$, C$
30 DATA 1, 2, 3, "ONE", "TWO",
"TRHEE"
40 PRINT  A, A$
50 PRINT  B, B$
60 PRINT  C, C$
```

```
70  END
```

运行后的结果为

```
1        ONE
2        TWO
3        THREE
```

如果我们将30行错打成

```
30  DATA  1, "ONE", 2, "TWO", 2,
"THREE"
```

执行后屏幕上会出现如下的信息：

```
ERR  CODE  9
AT   LINE  10
```

这说明变量和所需的数据发生了不对应的错误。变量B读取了字符串ONE，这是不允许的。

如果数据区中的数据不够也会引起停机，但数据多了一般不会影响程序的执行。

例 7.2 计算3.7，5.8，7.6，2.1这四个数的平均值。

编这个程序方法很多，这里我们仅列出两个供参考。

程序1

```
10  READ  A, B, C, D
20  M = (A+B+C+D)/4
30  PRINT "MEAN="; M
40  END
50  DATA  3.7, 5.8, 7.6, 2.1
```

运行后的结果为

```
MEAN=4.8
```

如果我们在50行少送一个数据

```
50  DATA  3.7, 5.8  2.1
```

运行时屏幕上就会出现如下的信息：

```
ERR  CODE  32
AT  LINE  10
```

这表示在第10行的READ语句中，由于变量读不到数据而导致停机。

程序2

```
10  S=0
20  READ  A
30  IF  A=-1  THEN  GOTO  60
40  S=S+A
50  GOTO  20
60  M=S/4
70  PRINT "MEAN=";  M
80  END
90  DATA  3.7, 5.8, 7.6, 2.1, -1
```

这个程序采用了循环求和及标志转移的方式。第40行是典型求和语句，而数据-1是转移标志，这种方式在程序设计中经常用到。

四、特殊DATA语句

一般情况下，DATA语句中的数据都是常数，但COMX—PC1电脑还允许DATA语句中使用变量。例如

```
10  READ  A, B, C
20  DATA  3, 5
30  I=4
```

```
40  DATA  I
50  I=A
60  READ  A, X
70  DATA  A, I
80  PRINT  A, B, C, I, X
90  END
```

执行后结果为

```
3     4     5     3     3
```

40行中I的值为4。70行中A的值为3，I的值也为3。

§2 COMX—PC 1 电脑的音响功能

COMX-PC1电脑具有相当强的音响功能。电脑靠某些指令控制音响合成器推动扬声器发出声音。

一、音乐合成音

MUSIC是一个音乐合成音指令。通过该指令，电脑可以奏出优美动听的音乐。MUSIC的格式为：

行号 MUSIC (X, Y, Z)

参数X决定音符，可取值1，2，3，4，5，6，7，表示一个八度内的“DO”，“RAY”，“ME”，“FAR”，“SO”，“LA”，“SEE”。参数Y决定音程，共有8个音程，分别取值1，2，3，4，5，6，7，8，值越大音调越高。参数Z决定音量，共有15个响度，分别取值1，2，3，4，5，6，7，8，9，10，11，12，13，14，15，数值越大声音越响。

MUSIC指令可以立即执行。一旦执行了MUSIC指令，

扬声器就一直发声，让它停止发声需输入以下指令：

MUSIC（0，0，0）

二、噪　音

COMX-PC1电脑可以发出各种噪音，这些噪音常用于游戏程序中模拟某些声响，以增强效果。噪音指令为NOISE，它的格式为：

行号　NOISE（Y，Z）

参数Y决定噪音的频率，可取值1，2，3，4，5，6，7，8，数值越大频率越高。参数Z决定音量，音量分为15级，Z取值范围为1至15，数值越大声音越响。

三、单　音

COMX-PC1电脑有个频率发声装置。利用单音指令TONE可以提供准确的音频。TONE指令的格式为：

行号　TONE（X，Y，Z）

参数X决定声音的频率，取值范围为1至128，各个数值表示不同的频率。参数Y决定音程，取值范围为1至8，即有8个音程。X，Y参数结合能产生8×128个频率。参数Z的作用与MUSIC，NOISE中相应参数的作用相同。表7.1列出了TONE指令中参数X与实际输出频率之间的关系，它对应于Y＝4的情况。若TONE中参数X不变，输出频率与音程成正比。即每高一个音程，输出频率增加一倍，每低一个音程输出频率减少到原来的1／2。

用单音配合MUSIC指令可以奏出各种各样的曲子。此外，为了使音响功能更加逼真，COMX—PC1电脑还提供了大范围音量调节指令。其格式为：

行号　VOLUME（Z）

表7.1　TONE指令中参数X与频率的关系（Y = 4）

参数X	频率（Hz）	音　符	参数X	频率（Hz）	音　符
86	257	C	82	270	$D^b/C^\#$
77	287	D	73	303	$E^b/D^\#$
69	321	E	65	341	F
61	363	$G^b/F^\#$	58	382	G
54	410	$A^b/G^\#$	52	426	A
49	452	$B^b/A^\#$	46	482	B

参数Z可取值 1， 2， 3， 4，数值越大音量越大，该指令与前面三个指令（尤指其中的Z参数）配合，可以产生60个不同级别的响度。

四、等　待

为了编曲的需要，COMX—PC1电脑专门提供了等待语句WAIT，它可以使电脑在运行中等待一段时间。WAIT指令格式为：

行号　WAIT（表达式）

表达式的值必须为正整数，数值 1 相当电脑CPU时钟128000。WAIT（1）相当于暂停约6.4毫秒，这个指令不单用于编曲，而且在提示程序中也常用到。

例 7.3　编一程序，由COMX—PC1电脑演奏贝多芬

的第九交响乐的第四乐章主题“欢乐颂”。

简谱如下：

1＝D 4/4

3 3 4 5 | 5 4 3 2 | 1 1 2 3 | 3· 2 2 — |

3 3 4 5 | 5 4 3 2 | 1 1 2 3 | 2· 1 1 — |

2 2 3 1 | 2 34 3 1 | 2 34 3 2 | 1 2 5 3 |

3 3 4 5 | 5 4 3 42 | 1 1 2 3 | 2· 1 1 — |

程序如下：

```
10  READ  X, Y
20  IF  X=0  THEN  TONE (52, 3, 3):
GOTO 60
30 IF X=-1 THEN TONE (0, 0, 0): END
40  IF  X=A  THEN  TONE (X, 4, 1)
50  TONE (X, 4, 3)
60  WAIT (T*80)
70  A=X: GOTO  10
100  DATA  61, 1, 61, 1, 58, 1, 52, 1,
     52, 1, 58, 1, 61, 1, 69, 1
110  DATA  77, 1, 77, 1, 69, 1, 61, 1,
     61, 1.5, 69, .3, 69, 2.4
```

```
120 DATA 61, 1, 61, 1, 58, 1, 52, 1,
    52, 1, 58, 1, 61, 1, 69, 1
130 DATA 77, 1, 77, 1, 69, 1, 61, 1,
    69, 1.5, 77, .3, 77, 2.4
140 DATA 69, 1, 69, 1, 61, 1, 77, 1,
    69, 1, 61, .3, 58, .3, 61, 1, 77, 1
150 DATA 69, 1, 61, .3, 58, .3, 61, 1,
    69, 1, 77, 1, 69, 1
160 DATA 0, 1, 61, 2.4
170 DATA 61, 1, 58, 1, 52, 1, 52, 1,
    58, 1, 61, 1, 58, .3, 69, .3
180 DATA 77, 1, 77, 1, 69, 1, 61, 1,
    69, 1.5, 77, .3, 77, 2.4
190 DATA -1, 0
```

该程序给出用单音奏出的D调乐曲，音程为5，因为只有一个低音部的音符，故在20行加入了一个判断，在数据中加了标志“0”，这样就将参数Y固定了。数据“-1”是停机标志。第70行中的A=X和40行的作用是为了使非连音的同音符产生节奏，如果没有这些语句，就会奏出连音。需要说明的是，程序中采用了固定音量，这是不符合乐理的，因而该程序产生的乐曲失去了原旋律固有的抑、扬、顿、挫的韵味，这无疑是大煞风景。若按实际规律处理强弱音量，数据量将要大大增加。有兴趣的读者可以自行尝试。

§3　其它的语句和函数

COMX—PC1 电脑还具有一些特殊语句和函数，有些是其他 BASIC 语言没有的。

一、恢复数据区语句

在程序中我们常常用到一些重复的数据。这就需要用RESTORE指令。该指令的格式为：

行号　RESTORE

RESTORE 指令用在有READ、DATA语句的程序中，该指令可使数据区指针恢复到数据区的开始位置。

例 7.4 下面给出一个关于使用RESTORE指令的程序示例。

```
10  READ  A, B, C
20  READ  X, Y, Z
30  DATA  1, 2, 3, 4, 5, 6
40  RESTORE
50  IF  X=4  THEN  GOTO  20
60  PRINT  A, B, C
70  PRINT  X, Y, Z
80  END
```

运行后屏幕上显示为：

```
1          2          3
1          2          3
```

程序执行第10、20行后，变量A，B，C，X，Y，Z中的数值分别是1，2，3，4，5，6。第40行语句使数据区的

指针回到开始的位置。由于X＝4，所以执行20行，此时X，Y，Z的值分别变为1，2，3了。

二、几个函数

1．MOD函数

MOD函数是求两个数值相除的余数。它的形式是

MOD（X，Y）

两个参数X，Y均可以是表达式、变量、常数。MOD函数的实际计算过程为

X－（X／Y）＊Y

其中每个参数都需先通过四舍五入化为整数，给出的结果也为整数。例如

```
PRINT MOD (10, 3)
```

结果是1。又如

```
PRINT MOD (101.4, 33.7)
```

结果是33。

2． INUM 函数和 FNUM 函数

INUM 和 FNUM 是一对函数。它们都只有一个参数，一般形式为：

INUM（X）

FNUM（X）

INUM的功能是将浮点数转化为整型整数，标准是进行四舍五入，取其最接近的整数。例如：

```
PRINT INUM (7.8)
```

结果为8。又如：

```
PRINT INUM (7.4)
```

结果为7。

INUM 函数称为整型函数。

注意：INUM函数不同于INT函数。INT函数称为取整函数，它取整时并不进行四舍五入，并且取整后仍为浮点型整数。即INT（X）自动给出一个不大于X值的最大整数，例如：

INT（-7.6）=-8，INT（7.6）=7，这里-8与7都是浮点型整数。

INUM（—7.6）=-8；INUM（7.6）=8，这里8与-8均为整型整数。

FNUM的作用是使整型整数转化为浮点型整数。例如：

PRINT FUNM （INUM（7.9））

结果是8。但这个8是浮点型的整数。

三、调试程序的一些命令的用法

1．LIST命令

LIST命令在前面已经多次用到了。电脑屏幕上只能显示22行字符。如果我们想看其中的若干行，可以借助LIST命令来实现。

LIST 行号n_1，行号n_2

这种形式是让电脑列出从行号n_1至行号n_2之间的全部程序。但要注意，行号n_1的值应小于行号n_2的值。当然使用这种形式不能列出多于22行的程序。

2．RENUMBER命令

前面已经提到过可使用RENUMBER命令来调整行号。在使用时应该注意程序中的GOTO语句。如果程序中有

30 GOTO 200 语句，

执行RENUMBER命令后，行号200会自动相应地改变。而

对于

30 GOTO 4＊50 语句，

执行RENUMBER命令后，仍然会是GOTO 4＊50，这样就可能影响你的程序，破坏了它的正常运行。

3． RUN命令

RUN 命令我们已经很熟悉了，但在调试程序时，如果只需要运行某一部分，我们利用如下命令：

RUN 行号

该命令可以从指定的行号开始运行程序，显然会使调试工作节省时间。

§4 综合例题

例 7.5 编一程序，让 COMX—PC1 电脑奏出歌曲“我的中国心”。可不考虑声音强弱。

歌曲简谱如下：

1＝^{b}E 4／4

(3—6－| 1 76 31 | 6－－－| 6－－－|3－3－| 2 32 1 7 |

6－－－| 6－－0)|

6·3 23 17 | 6－－0 | 3 6 5 32 12 | 3－－35 |

6·7 65 32 | 1 12 3－ | 2·3 76 5 | 6－－0 |

6·3 23 17 | 6－－0 | 3 6 5 32 12 | 3－－35 |

9·7 65 32 | 1 1 2 3— | 2·3 76 5 | 6— —03 |

:5·3 3 03 | 1·6 6 61 | 6 5 1 2 12 | 3— — 03 |

1·6 6·6 | 1·2 3— | 3 32 7·5 | 6 — —0 |

6·3 23 17 | 6— —0 | 3 6 5 32 12 | 3— — 35 |

6·7 65 32 | 1 1 2 3— | 2·3 76 5 | (1) 6— —(0 3 5 6 |

1·7 7·6 | 6·3 3— | 6·4 4 3 | 2— —0 |

7·6 6·5 | 5·2 2— | 03 12 3·7 | 6— —) 03 :‖

(2) 6— — — | 6— —0 ‖

程序为:

```
10  J=9 : I=1
20  GOTO  50
30  RESTORE : I=—1 : J=18
40  READ  A, B, C, A, B, C, A, B, C,
45  IF  C< >  —1  THEN  GOTO  40
50  READ  X, Y, T
60  IF  X=0  THEN  MUSIC (0, 0, 0) :
    GOTO  110
70  IF  X=Z  THEN  MUSIC  (X, Y, 1)
```

```
80  Y=Y--1
90  MUSIC (X, Y, 4)
100 IF  T=-2  THEN  GOTO  30
110  WAIT (T*30)
120  IF  T=18  THEN MUSIC(0, 0, 0): END
130  Z=X
140  GOTO  50
150  DATA 3, 5, 6, 6, 5, 6, 1, 6, 3,
     7, 5, 1, 6, 5, 1
160  DATA 3, 6, 3, 1, 6, 3, 6, 5, 20,
     3, 5, 6, 3, 6, 6
170  DATA  2, 6, 3, 3, 6, 1, 2, 6, 1,
     1, 6, 3, 7, 5, 3
180  DATA  6, 5, 17, 0, 0, 3, 6, 4, 4,
     3, 5, 1, 2, 5, 1
190  DATA  3, 5, 1, 1, 5, 1, 7, 4, 1, 6,
     4, 9, 0, 0, 3
200  DATA  3, 5, 1, 6, 5, 3, 5, 5, 1,
     3, 5, 1, 2, 5, 1
210  DATA  1, 5, 1, 2, 5, 1, 3, 5, 9,
     3, 5, 1, 5, 5, 1
220  DATA  6, 5, 4, 7, 5, 1, 6, 5, 1,
     5, 5, 1, 3, 5, 1
230  DATA  2, 5, 1, 1, 5, 1, 1, 5, 3,
     2, 5, 1, 3, 5, 6
210  DATA  2, 5, 4, 3, 5, 1, 7, 4, 1,
```

```
    6, 4, 1, 5, 4, 3
250 DATA 6, 4, 9, 0, 0, 3, 6, 4, 4,
    3, 5, 1, 2, 5, 1
260 DATA 3, 5, 1, 1, 5, 1, 7, 4, 1,
    6, 4, 9, 0, 0, 3
270 DATA 3, 5, 1, 6, 5, 3, 5, 5, 1,
    3, 5, 1, 2, 5, 1
280 DATA 1, 5, 1, 2, 5, 1, 3, 5, 9,
    3, 5, 1, 5, 5, 1
290 DATA 5, 5, 4, 7, 5, 1, 6, 5, 1,
    5, 5, 1, 3, 5, 1
300 DATA 2, 5, 1, 1, 5, 1, 1, 5, 3, 2,
    5, 1, 3, 5, 6
310 DATA 2, 5, 4, 3, 5, 1, 7, 4, 1,
    6, 4, 1, 5, 4, 3
320 DATA 6, 4, 9, 0, 0, 1
330 DATA 3, 5, 1, 5, 5, 4, 3, 5, 1,
    3, 5, 3, 0, 0, 1
340 DATA 3, 5, 1, 1, 6, 4, 6, 5, 1,
    6, 5, 3, 6, 5, 1
350 DATA 1, 6, 1, 6, 5, 3, 5, 5, 3,
    1, 5, 3, 2, 5, 1
360 DATA 1, 5, 3, 2, 5, 3, 3, 5, 9,
    0, 0, 1, 3, 5, 1
370 DATA 1, 6, 4, 6, 5, 1, 6, 5, 4,
    6, 5, 1, 1, 6, 4
```

```
380 DATA  2, 6, 1, 3, 6, 9, 3, 6, 1,
     2, 6, 1, 7, 5, 4
390 DATA  5, 5, 1, 6, 5, 9, 0, 0, 3,
     6, 4, 4, 3, 5, 1
400 DATA  2, 5, 1, 3, 5, 1, 1, 5, 1,
     7, 4, 1, 6, 4, 9
410 DATA  0, 0, 3, 3, 5, 1, 6, 5, 3,
     5, 5, 1, 3, 5, 1
420 DATA  2, 5, 1, 1, 5, 1, 2, 5, 1,
     3, 5, 9, 3, 5, 1
430 DATA  5, 5, 1, 6, 5, 4, 7, 5, 1,
     6, 5, 1, 5, 5, 1
440 DATA  3, 5, 1, 2, 5, 1, 1, 5, 1,
     1, 5, 3, 2, 5, 1
450 DATA  3, 5, 6, 2, 5, 4, 3, 5, 1,
     7, 5, 1, 6, 5, 1
460 DATA  5, 5, 3, 6, 5, 1
470 DATA  0, 0, 1, 3, 5, .1, 5, 5,
     .1, 6, 5, .1
480 DATA  1, 6, 4, 7, 5, 1, 7, 5, 4,
     6, 5, 1, 6, 5, 4
490 DATA  3, 5, 1, 3, 5, 6, 6, 4, 4,
     4, 5, 1, 4, 5, 3
500 DATA  3, 5, 3, 2, 5, 9, 0, 0, 3,
     7, 5, 3, 6, 5, 1
510 DATA  6, 5, 4, 5, 5, 1, 5, 5, 4,
```

```
    2, 5, 1, 2, 5, 6
520 DATA  0, 0, 1, 3, 5, 1, 1, 5, 1,
    2, 5, 1, 3, 5, 4
530 DATA  7, 5, 1, 6, 5, -2
```

该程序解释如下：

本曲较为复杂，没有用混合方式按降E调编写，只是采用了通普的MUSIC指令。

第30、40行及50行是为了使重复时加快空读数据速度而特意设置的。第80行使音程Y降一级，如想听高八度的曲子可取消这一行。

这个程序在数据中采用了一些技巧，其中最后一行的“-2”是重复标志。变量I，J是数值变换标志。开始的J值为9，I值为1，具有等待时间作用。重复后的I值为—1．做为跳出空读的标志，而J值为18，做为重复后音符演奏时间参数及停机标志。

该程序并不是一个理想的程序，待今后学习循环、数组等知识后，编排这类程序会更加方便，同时结构上也会有所改进。

§5 练习题

一、试编一个程序，让COMX—PC1电脑演奏一首自己喜爱的歌曲。

二、用READ、DATA语句编写一个计算平均成绩的程序。

第八章　循环与数组

§1　循　环

在第六章中介绍了条件语句和转向语句，利用它们可以实现指定次数的循环。一般来说，循环都有共同点：一是有初值；二是有步长；三是有终值判断。为了使循环的结构更加清楚，BASIC 语言里专门设置了循环语句 FOR—NEXT。

一、循环的结构

循环的结构由循环语句和循环体构成，循环语句由说明语句和终止语句构成。

1．循环说明语句

FOR语句即循环说明语句，FOR语句的格式为：

行号 FOR 变量=表达式 1 TO 表达式 2 STEP 表达式 3 其中FOR，TO，STEP都是BASIC保留字。变量是指简单变量，它是控制循环的主要因素。表达式 1 为循环变量的初始值，表达式 2 为循环变量的终止值，表达式 3 为循环变量的步长值。当此值为 1 时，STEP保留字是可以省略的。例如

```
10 FOR I = 1  TO 100 STEP  1
```

与

```
10 FOR I = 1  TO 100
```

是等价的。

2．循环终止语句

NEXT 语句是循环终止语句。NEXT语句的格式为：

行号 NEXT 变量

这里的变量一定要与相应的FOR语句中的循环变量一致。尽管NEXT语句是循环终止语句，但并不是执行到它就一定停止循环，终止循环是有条件的。

3．循环体

FOR语句和NEXT语句之间的全部可执行的BASIC语句叫做循环体。循环体本身不包括FOR和NEXT语句。

下面用一个例子来具体说明循环语句的作用。

例 8.1 打印出 1 至10之间的整数及它们的总和。

程序如下：

```
10  S = 0
20  FOR I = 1  TO 10
30  S = S + I
40  PRINT I
50  NEXT I
60  PRINT S
70  END
```

第20行表示循环变量初始值为 1 ，终止值为10，步长为 1 。第50行是NEXT语句，表示循环到此终止。第30行、第40行为循环体。

程序开始执行10行，首先将S的值赋 0 。执行 20 行，将 1 赋给循环变量I，执行30、40两行后输出了I 的值，这时I 值为 1 。执行50行，将I的值增加 1 ，这时I的值是 2 ，然后与终值10进行比较，看是否大于终值。如果不大则继续执行循

环，如果大于终值则执行循环以外的语句。I 值为 2 不大于10，所以继续执行30、40行。至 50行I 的值又增加 1 。再去比较，依此类推，直至 I 的值为11时，才能去执行第60行，打印出 1 至10之间的整数和。此程序执行后的结果为：

```
1    2    3    4    5
6    7    8    9    10
55
```

二、循环的用法

1．使用循环时，FOR语句和NEXT 语句必须成对出现，缺一不可。

2．循环的次数等于循环变量的终值减去循环变量的初值除以步长，取整后再加 1 。

3．用FOR语句不限制初值、终值、以及步长的正、负。但要注意当步长为负数时，循环变量的值要小于终值才能完成循环。凡是不符合循环规定的程序，循环只进行一次，然后就会发生错误。（有的电脑将继续执行下边的语句）。

三、循环嵌套

循环嵌套也叫多重循环。即在循环之中还有循环。从整体结构看，内循环又是外循环的循环体了。

例 8.2　利用双重循环打印一个三角形。

程序如下：

```
10  INPUT "N=" N
20  FOR I=1 TO N
30  FOR J=1 TO I
40  PRINT "*";
50  NEXT J
```

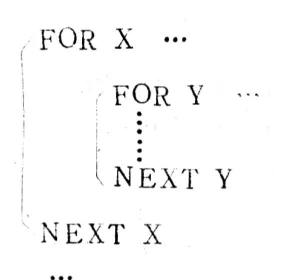
```
60  PRINT
70  NEXT I
80  END
```

程序中的N决定三角形的大小，其中40行为内循环的循环体，30至60行是外循环的循环体。

在使用多重循环时，特别要注意以下几点：

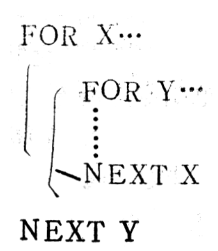
1．内外循环不得交叉

比如



是正确的。

而



是错误的。

2．多重循环中可以使用条件语句

在循环中使用条件转移语句是允许的。不过，在多重循环中使用条件转移语句一定要注意：不能从外循环转入内循

环。但从内循环转到外循环是允许的。例如下面的程序：

```
10  FOR  I=1  TO  10
20  FOR  J=1  TO  10
30  PRINT  I, J
40  IF  I=3  THEN  GO  TO  60
50  NEXT  J
60  NEXT  I
70  END
```

是正确的。而程序

```
10  FOR  I=1  TO  10
20  FOR  J=1  TO  10
30  PRINT  I, J
40  NE×T  J
50  IF  I=7  THEN  GOTO  30
60  NEXT  I
70  END
```

是错误的，它的第50行可能使程序转移到内循环中去。

上面的程序中使用了转到循环之外的条件语句。实际上，COMX—PC1 电脑专门设置了 EXIT 指令用来代替 GOTO指令，从而清楚地表示转出循环。例如：

```
10  FOR  I=1  TO  10
20    FOR  J=1  TO  I
30    PRINT "*";
40    IF  J>3  THEN  EXIT  60
50    NEXT  J
60  PRINT
```

```
70  T=I+J
80  PRINT  T
90  NEXT  I
100 END
```

第40行表示当J的值大于 3 后去执行第60行。

循环在程序设计中用处很多，为了提高效率，在编写程序时要注意尽量减少循环次数。

§ 2　数组和字符串组

一、数组

数组是由一组具有同名的下标变量组成。

1．下标变量

下标变量是由简单变量加上可以改变的下标组成。比如，A（3），B（6,5）都是下标变量。下标变量分为单下标变量（例如 A(5)，B1(7)等）和双下标变量（例如 C（7,2），D（12,7）等）。

2．数组

由同名的单下标变量组成的数组叫一维数组。由同名的双下标变量组成的数组叫二维数组。一维数组中，每个元素（下标变量）只有一个参数。而二维数组中，每个元素都有行标和列标两个参数。从数据结构的观点看，参数（下标）决定了元素在数组中的位置。

3．数组说明

数组元素的个数由数组说明语句DIM来确定，数组元素个数的说明一定要准确，少了不够用，多了浪费内存。一

般每个元素占有4个字节，而数组名占用5个字节。比如用：

DIM　A（10，10）

说明两维数组A，含有100个元素，一共占有400个字节，再加上5个字节的名字，共占405个字节。又如

DIM　B（30）

说明一维数组B，包含30个元素，占有125个字节。

DIM语句的一般格式为：

行号 DIM 数组名（表达式）

行号 DIM 数组名（表达式1，表达式2）

式中表达式的值允许有多大呢？COMX—PC1电脑允许一维数组的元素个数最多为255个。对二维数组元素个数的限制则视程序占用内存多少而定，也就是说如果你所编的程序很长，那么你在这个程序中所用的两维数组元素的个数就要相对的减少，因为机器的内存是一定的；如果你所编的程序较短，那么你在这个程序中所用的两维数组元素的个数就可以相应的增加。

如果说明数组时超出规定的范围，电脑会给出溢出错误信息。

4．数组类型

根据数组名的类型，数组可分为浮点型和整型两种。如果变量A至Z都是浮点型，那么相应的数组也是浮点型。例如A被定义为整型变量，那么A数组也是整型数组。不同名称、不同类型的数组可以放在同一行内说明。例如

15　DIM　A（10，10），B（30），C（25）

5．说明语句的省略

通常，程序中数组语的元素少于10个则可以省略说明语

句。在电脑中，已经留出一定的内存供没有说明的数组存放数据。

6．清除数组数据区

执行CLD命令或对程序进行编辑后，数组的数据区都会被清除。因此，当不需要数据空间时，可以用CLD命令来清除数据空间。

二、字符串组

COMX—PC1电脑具有处理字符串组的能力。字符串组同数组的形式基本一样，只是需要在数组名后加上“$”符号。例如，A$（4），B$（3，6）等都是字符串组。下标值的限制同数组一样。

字符串组不需要说明，只须将字符串赋给字符串组即可。如果字符串组的下标变量没有实际内容，是不能用的。例如：

```
10  A$ (10) = "A"
20  A$ (7) = "B"
30  PRINT  A$ (10), A$ (7)
40  END
```

执行后的结果为：

```
A        B
```

这个程序中只有A$(10)和A$（7）存在，而A$（3），A$（6）……是不存在的。在字符串组数据空间中也没有给A$（3），A$（6）等留内存空间。

CLD和编辑命令同样能清除字符串组数据空间。

总之，字符串组同数组的用法基本一样。只是要注意数组需要说明。

三、数组的赋值

使用数组一定要给它们赋值。赋值之前，往往需要清零，清零一般采用循环赋零值的方法。下面将举例说明。

对于一维数组常用下述两种方法清零：

1．

```
10  DIM  A(30), B(30)
20  FOR  I=1  TO  30
30  A(I) = 0
40  B(I) = 0
50  NEXT  I
60  END
```

2．

```
10  INPUT  "N=" N
20  DIM  A(N), B(N)
30  FOR  I=1  TO  N
40  A(I) = 0
50  B(I) = 0
60  NEXT  I,
```

对于二维数组可用下面的方法清零：

```
10  DIM  A(5, 50), B(5, 50)
20  FOR  I=1  TO  5
30    FOR  J=1  TO  50
40            A(I, J) = 0
50            B(I, J) = 0
60  NEXT  J
70  NEXT  I
```

```
80  END
```

给数组赋值多采用循环读值的方法。例如下面的一维数组要读30个值。

```
10  DIM  A (30)
20  FOR  I = 1  TO  30
30  READ  A (I)
40  NEXT  I
100  DATA  5, 6, 8, …      28
           \_________________/
                  30个
```

这种赋值是通过循环变量 I 的变化给A数组不同元素赋值来实现的。同理，二维数组赋值也可采用这种方式，只不过需增加一重循环。例如：

```
10  DIM  C (10, 25)
20  FOR  N = 1  TO  10
30    FOR  M = 1  TO  25
40    READ  C (N, M)
50    NEXT  M
60  NEXT  N
100 DATA  5, 10, 31, …  92
          \______________/
               250个
```

以上可以看出，无论是清零还是赋值，一维数组采用单循环法，二维数组采用双重循环法。

当需要将大量数据进行排序的时候，采用数组方法非常方便。下面列举两个排序的程序，数据可 以 由 随 机 数 给出。

程序 1

```
10  DIM  A (20)
20  FOR  I = 1  TO  20
30  A (I)  = INT (10 + RND * 100)
40  NEXT  I
50  FOR  I = 1  TO  19
60      FOR  J = I + 1  TO  20
70      IF  A (I) > = A (I)  THEN  GOTO 90
80      T = A (I) : A (I) = A (J) : A (J) = T
90      NEXT  J
100  NEXT  I
110  END
```

第50行至100行为排序部分。顺序为从大至小。

程序 2

```
10  INPUT  N
20  DIM  A (N)
30  FOR  I = 1  TO  N
40  A (I)  = INT (10 + RND + 100)
50  PIRNT  A (I) ,
60  NEXT  I
70  FOR  I = 1  TO  N - 1
80      FOR  J = I + 1  TO  N
90      IF  A (I) > = A (J) THEN GOTO  110
100     T = A (1) : A (1) = A (J) : A (J) = T
110     NEXT  J
120 PRINT  A (I)
130 NEXT  1
```

```
140 END
```

第70至130行为排序部分。第50行、120行为输出结果。排序的方法很多，这里就不再列举了。

§3 综合例题

例 8.3 编一程序，由电脑给出100至200之间的16个不同的随机整数。将这些数中的奇数按从大到小的顺序排列，偶数按从小到大的顺序排列，并将排序结果打印，最后分别打印出奇数和偶数的个数。

本题是一道综合知识的考查题，不仅考查随机函数的概念和选取16个不同随机数的方法，而且考查数组的清零、赋值及排序方法，同时还考查偶数的选取、计数以及使用打印语句的灵活性。下面列出一个程序供参考。

```
10  DIM  A (16) , B (16)
50  X=0 : Y=0
60  FOR  I=1  TO  16
70  A (I) =INT (100*RND+100)
80  IF  I=1  THEN  GOTO  120
90      F R  J=1  TO  I-1
100     IF  A (I) =A (J)  THEN  EXIT  70
110     NEXT  J
120  PRINT  A (I),
130  NEXT  I
140  PRNT : PRINT
```

```
150 FOR  I=1  TO  15
160     FOR  J=I+1  TO  16
170     IF  A (I) >A (J)  THEN GOTO 190
180     B=A (I) : A (I) =A (J) : A (J) = B
190     NEXT  J
200 NEXT  I
210 FOR  K=1  TO  16
220 IF  A (K) /2< >INT (A(K)/2)  THEN
    PRINT  A(K): X=X+1 : GOTO  250
230 Y=Y+1
240 B (Y) =A (K)
250 NEXT  K
260 FOR  I=Y  TO 1 STEP  -1
270 PRINT  B (I)
280 NEXT  I
290 PRINT
300 PRINT "JI  SHU: "; X
310 PRINT "OU  SHU: "; Y
320 END
```

这个程序60至130行给A数组赋值。其中内循环90至110行可保证取的数不相等。150至200行为A数组排序过程。210至250行分开奇偶数，其中220行打印出奇数并计数，230行对偶数计个数。260至280行打印出偶数。300行及310行分别打印出奇偶数的个数。

例 8.4 学生成绩分析

某班有N个学生参加某门功课考试，

要求统计考试分数100分的有多少人？90至99分的有多少人……60至69分的有多少人？不及格的有多少人？

程序

```
10  INPUT  N
20  DIM  A (N) , S (10)
30  FOR  I = 1  TO  N
40  READ A (I)
50  NEXT  I
60  FOR  J = 1  TO  10
70  S (J) = 0
80  NEXT  J
90  FOR  I = 1  TO  N
100 IF A (I) <60 THEN S (5) =S (5) + 1
    : GOTO  130
110 R=INT (A(I)/10)
120 S (R) =S (R) + 1
130 NEXT  I
140 PRINT "100" , S (10)
150 FOR  I = 9  TO  6  STEP  - 1
160 PRINT I * 10; "---" ; I * 10+ 9 , S(I)
170 NEXT  I
180 PRINT "0 --- 59" , S (5)
190 END
200 DATA  65, 57, 61, 71, 75, 80, 93, 91
210 DATA  82, 51, 32, 97, 100, 100, 92
...
```

程序中90至130行为统计计数部分。利用下标值为参考量，这种方法用途很广。程序中的数据不能少于N个。

例 8.5 某校对M个学生进行一次测验，共测验五道题，每题有三个选择答案。规定每个人对每题只能选择一种答案。求出每题中选择每种答案的人数各是多少。

```
10  DIM  S（5， 3）
20  FOR  I=1  TO  5
30      FOR  J=1  TO  3
40      S（I， J）=0
50      NEXT  J
60  NEXT  I
70  INPUT "M=" M
80  FOR  N=1  TO  M*5
90  READ  Q， R
100 S（Q， R）=S（Q， R）+1
110 NEXT  N
120 FOR  I=1  TO  5
130     FOR  J=1  TO  3
140     PRINT  I， J， S（I， J）
150      NEXT  J
160   PRINT
170  NEXT  I
180  DATA  1， 2， 2， 1， 3， 2， 5， 1
190  DATA  …
…
```

程序中变量Q为题目号，R为答案号。DATA中的数据要一

对一对地给，例如第一题答案2，就给出1，2。其他类推，这个程序就是利用二维数组的下标来完成统计工作的。总数据量应该有10M个。

例 8.6 假如某班有45名学生，期末进行六门功课的考试。要求：

1．统计每个学生的各科成绩、总分、平均分、并连同学生姓名一起以表格形式输出（表1）。

2．总分按名次排列并连同姓名一起以表格形式输出（表2）。

学生数及科目数分别用N及Q表示。Y数组存放学号，A$字符串组存放姓名。N数组存放各科分数及总分、平均分。S数组存放排名次后的总分。程序如下：

```
10  INPUT  N, Q
20  DIM  Y (N) , N (N, Q+2) , S (N)
30  FOR  I=1  TO  N
40  READ  Y (I) , A$ (I)
50      FOR  J=1  TO  Q
60      READ  N (I, J)
70      NEXT  J
80  NEXT  I
90  PRINT
100 FOR  I=1  TO  N
110  S (I) =0
120     FOR  J=1  TO  Q
130      S (I) =S (I) +N (I, J)
140      NEXT  J
```

```
145  N (I, Q+1) =S (I)
150  N (I, Q+2) =INT (S(I)/Q*10) /10
  160  NEXT  I
170  FOR  I=1  TO  N-1
180      FOR  J=I+1  TO  N
190      IF S (I) >=S (J)  THEN GOTO 210
200      T=S (I) : S(I)=S (J) : S (J) =T
210      NEXT  J
220  NEXT  I
230  PRINT"I NAME Q1 Q2 Q3 Q4 Q5 Q6";
240  PRINT "SUM  AVER"
250  FOR  I1=1  TO  N
260  PRINT  Y (I1); " "; A$ (I1);
270      FOR  J1=1  TO  Q+2
280      PRINT  TAB(4*J1+10); N(I1,J1);
290      NEXT  J1
300  PRINT
310  NEXT  I1
315  PRINT  "N", "NAME", "SUM"
320  FOR  I1=1  TO  N
330      FOR  J1=1  TO  N
340      IF S(I1)<>N(J1,Q+1) THEN GOTO
350      PRINT  I1, A$ (J1), S (I1)
360      NEXT  J
370  NEXT  I
380  DATA 5301, "LI MING",65,71,82,90,68,
```

```
       92
390  DATA  5302, "WANG  FENG", 80,
       90, 100, 98, 92, 96
400     …
…
1000  END
```

这个程序的30至80行读入原始数据。100至160行求总分和平均分。170至220行进行总分排序。230至310行打印表1,315至370行打印表2,从380行起输入原始数据。

§4 练 习 题

一、对给定的数据 1,2,4,3,4,2,1,2,3,2,1,1,3 求出其中有几个1?几个2?几个3?几个4?并求出它们各自与总个数的百分比。

二、结合一个班的具体情况编写学生成绩统计分析程序。

三、由电脑给出20个不同的300至400之间的随机整数。先打印出这些数。然后按最大的数,最小的数,次最大数,次最小数……打印出来。

四、打印一杨辉三角形

1

1 1

1 2 1

1 3 3 1

1 4 6 4 1

1 5 10 10 5 1

1 6 15 20 15 6 1

五、编写一个程序，将一已知数组中的各元素的值移到前一个元素中，而第一个元素中的值移到最后一个元素中。

第九章　子程序和实时控制

§1　子　程　序

在解决实际问题中，往往需要反复做只是其中某些参数不同的同样工作。BASIC的子程序提供了这种方便，就是把需要重复使用的程序段编成一个子程序。子程序有两类：一类是相同功能的程序段；另一类是各种不同功能的子程序。为了使整个程序结构清晰、合理、效率高、易懂易读，经常采用建立子程序的方法。

子程序是一段独立的程序。调用子程序或转子程序，程序执行效果是相同的。子程序是相对于主程序而言的，调用子程序的程序段称为主程序。

一、GOSUB　语句和RETURN语句

1．GOSUB　语句

GOSUB　语句称为调用子程序语句或转子程序语句。它的基本格式为：

行号　GOSUB　表达式

式中表达式的值为子程序开始的行号。

例如：

100 GOSUB　1000

表示程序执行到100行时，调用起始行号为1000的子程序。

或者说是转去执行1000行开始的子程序。

2． RETURN语句

RETURN语句称为子程序结束语句或返回语句。它 的格式为：

行号 RETURN

例如：

1050 RETURN

表示子程序执行到1050时结束，继而执行主程序。

例 9.1 编一程序，求三个整数的最大公约数。

编写此程序，关键是找二个整数的最大公约数。我们可以将求两个整数的最大公约数的过程写成子程序。

该子程序为：

```
1000 Q=INT (X/Y)
1010 R=X-Q*Y
1020 IF  R=0  THEN  GOTO  1060
1030 X=Y
1040 Y=R
1050 GOTO  1000
1060 RETURN
```

还可以写成：

```
1000 REM  SUBROUTINE
1010 Q=INT (X/Y)
1020 R=X-Q*Y
1030 IF  R=O  THEN  RETURN
1040 X=Y:Y=R
1050 GOTO  1010
```

这段子程序将RETURN直接写入1030行的条件语句中了。

我们现在来写主程序。

```
10  PRINT "THIS PROGRAM FINDS THE
    GCD"
20  PRINT
30  INPUT "A=" A
40  IF A<0 THEN END
50  INPUT "B=" B
60  INPUT "C=" C
70  X=A:Y=B
80  GOSUB 1000
90  X=C
100 GOSUB 1000
110 PRINT
120 GOTO 30
```

在主程序中40行为条件停机语句。当A的值小于零时，就会终止执行。80、100行两次调用子程序。

二、应用子程序时应该注意的问题

1．主程序中可以有多个调用子程序的语句，而且可以调用若干个不同的子程序。

2．子程序的行号一般应取得较大，且放于主程序之后。

3．除GOSUB语句外，不得以其它方式进入或调用子程序。

4．子程序中至少要有一个RETURN语句。子程序中使用多个RETURN语句也是允许的，其作用都是结束该段子程序。

5．调用子程序结束后，继续执行调用子程序语句下面的语句。

6．子程序允许再调用子程序，即子程序允许嵌套。嵌套多少要与循环语句一并考虑。一般不得超过8层。

§2 时间控制的子程序

COMX BASIC提供了TIMOUT和TIME语句，使程序执行经过某段时间后调用子程序。

TIMOUT和TIME语句的格式为：

行号 TIMOUT (X)

行号 TIME (Y)

X为子程序入口的开始行号数。Y为时间参数，每50为1秒钟。

TIMOUT与TIME必须一起使用。在程序中首先要说明TIMOUT，当执行到TIME时，开始计时，并继续执行下边的语句。经过由参数确定的时间后开始调用由TIMOUT决定的子程序。调用完成后，返回到TIMOUT之后的语句继续执行。直至下一次遇到TIME语句时再次计时。通常在这样的子程序中还设计另一个TIME语句来决定返回主程序的时间。这样就构成了由时间控制的子程序。

例 9.2 电子钟程序

在电脑屏幕上显示出一个数字电子钟，它应包括日历、时、分、秒，只要求时、分、秒变化即可。程序如下：

```
100 REM  DIGITAL  CLOCK
```

```
110 CPOS (0, 0) :CLS
120 CPOS  (10, 10)  :PRINT  "DIGITAL
    CLOCK"
130 WAIT (300)
140 CPOS (0, 0) :CLS
150 INPUT "YEAR:" Y1
160 INPUT "MONTH:" M1
170 IF  M1<1 THEN  GOTO  160
180 IF  M1>12  THEN  GOTO  160
190 INPUT  "DATE:" D1
200 IF  DI<1  THEN  GOTO  190
210 IF  DI>31  THEN  GOTO  190
220 INPUT  "HOURS:" H
230 IF  H>23  THEN  GOTO  220
240 INPUT "MINUTES" M
250 IF  M>59  THEN  GOTO  240
260 INPUT  "SECONDS:" S
270 IF  S>59  THEN  GOTO  260
280 FOR  I=1  TO  12
290 READ  M$ (I)
300 NEXTI
310 DATA  "JAN",  "FEB",   "MAR",
     "APR"
320 DATA  "MAY",  "JUN",  "JUL",
     "AUG"
330 DATA  "SEP",  "OCT",  "NOV",
```

```
    "DEC"
340 CPOS (0, 0) :CLS
350 CPOS (2, 18) : PRINT  Y1
360 CPOS (4, 10) : PRINT  M$ (M1)
370 CPOS (4, 15) : PRINT  D1
380 CPOS (10, 18) : PRINT ":"
390 CPOS (10, 21) : PRINT ":"
400 TIMOUT (1000)
410 GOSUB  2000
420 GOSUB  3000
430 GOSUB  4000
440 TIME (47)
450 E=KEY
460 IF  E<>  69  THEN  GOTO  450
470 CPOS (0, 0) :CLS
480 END

1000 TIME (47)
1010 S=S+1
1020 IF S<60 THEN GOSUB 4000: RETURN
1030 S=0
1040 GOSUB  4000
1050 M=M+1
1060 IF M<60 THEN GOSUB 3000: RETURN
1070 M=0
1080 GOSUB  3000
```

```
1090 MUSIC （2, 8, 5） : WAIT （2） :
     MUSIC (0, 0, 0)
1100 H=H+1
1110 IF H<24 THEN GOSUB 2000 : RETURN
1120 H=0
1130 GOSUB 2000
1140 RETURN

2000 IF H<10 THEN CPOS (10, 16) : PRINT
     " "; H : RETURN
2100 CPOS (10, 16) : PRINT H
2200 RETURN

3000 IF M<10 THEN CPOS (10, 19) :
     PRINT "  "; M : RETURN
3100 CPOS (10, 19) : PRINT M
3200 RETURN

4000 IF S<10 THEN CPOS (10, 22) : PRINT
     "  "; S : RETURN
4100 CPOS (10, 22) : PRINT S
4200 RETURN
```

程序的340至370行显示年、月、日。450行、460行为等待时间循环部分，一旦按下E键程序就会终止。第1090行为整点报时音响。子程序1000至1140行为时间控制子程序。子程序2000，3000，4000为时、分、秒显示子程序。当显示的

时间数字为1位时，在前面输出一个空格，然后再显示数字。（显示时间数字的变量为H、M、S）

§3 实时控制

在编写游戏程序时少不了进行实时控制。COMX—PC1电脑提供了 KEY 这个指令以满足实时控制的要求。KEY的基本格式为：

行号　**变量**=KEY

KEY是从键盘上直接寻找信息的指令。当执行 KEY 指令时，电脑通过键盘将字符的 ASCII 码值赋给变量。如果后边加入条件循环语句，即构成实时控制。例如

```
10  A=KEY
20  IF A< > 66 THEN GOTO 10
30  PRINT "KEY B PRESSED"
40  END
```

这个程序在第10行检查键盘，如果此时不按任何键则处于等待状态，如果按除B键以外的其他键（ESC、RT 除外）都将执行第10行，当B键按下后，就执行第30行，打印出：

KEY B PRESSED

注意，检查键盘只认一个键，即只读一个 ASCII 码。B键的ASCII码为66。全部字符的ASCII码列于书后的附录中。

编写游戏程序时，可以采用游戏杆的四个方向操作来控制动作，也可以利用方向键来控制动作。这四个键的ASCII码分别为上（136），右（137），下（138），左（139）。

例 9.3 活动图形程序

```
5   I=1  :  J=1
10  X=KEY
20  IF X=137 THEN GOTO 40
30  GOTO 10
40  CPOS (10, J) :PRINT "    "
50  CPOS (10, I) :PRINT "A"
60  J=I  :  I=I+1
70  IF  I=39  THEN  END
80  GOTO  10
```

该程序的功能是：按下右键即可打印出一个A字；再按一下右键，A字向右移动一格，……，这样连续按键就形成一个活动图形。此程序的关键是在字符A字位置上打印一个空格。

例 9.4 接球游戏

程序如下：

```
10  CPOS (0, 0) :CLS
20  Y=19:Z=0
30  DEFINT  X
40  X=RND (39)
50  B=1
60  FOR  K=1  TO  20
70  FOR  I=1  TO  20
80  A=KEY
90  IF  A=139  THEN  GOSUB  1000
100 IF  A=137  THEN  GOSUB  1100
110 IF X>39 THEN X=39:B=-B
```

```
120 IF X<0 THEN X=0：B=-B
130 CPOS (I, X) ：PRINT CHR$ (149)
140 CPOS (I, X) ：PRINT "    "
150 IF I=20 IF X=Y-1 THEN GOSUB 2000
160 IF I=20 IF X=Y THEN GOSUB 2000
170 IF I=20 IF X=Y+1 THEN GOSUB 2000
180 X=X+B
190 NEXT I
200 B=-B
210 NEXT K
220 END
1000 Y=Y-1
1010 IF Y=0 THEN Y=38：CPOS (20, 0) ：
     PRINT "    "
1020 CPOS (20, Y) ： PRIT CHR$ (32,
     148, 154, 32)
1030 RETURN
1100 Y=Y+1
1110 IF Y=37 THEN Y=0：CPOS (20, 37) ：
     PRINT "    "
1120 CPOS (20, Y) ： PRINT CHR$ (32,
     148, 154, 32)
1130 RETURN
2000 Z=Z+1
2010 CPOS (0, 38) ：PRINT Z
2020 TONE (52, 4, 5)
```

```
2030 TONE (0, 0, 0)
2040 RETURN
```

该程序在屏幕的第20行形成一个 |___| 形的“手”当球从屏幕上方投来时，操纵左、右键可以使“手”左右移动接球。接住一次记一分，积分在屏幕的右上角显示出来。

程序60行至210行构成整个游戏程序的核心。其中内循环为球从屏幕顶向下移动的过程。CHR$ 为字符串函数，第130行是印出ASCII码为149的图形（这里是一个球形）。ASCII码为148，154时构成 |___| 形。ASCII码为32是一个空格。子程序1000可使“手”向左移动，子程序1100可使“手”向右移动。子程序2000的作用是打印分数并发出声音。

§4 练 习 题

一、编写一个活动图形的游戏程序。设计几个点，随机从某处弹出一个球，球碰到点即计分，各点的分数可以不同。游戏进行到一定时间后停止，显示总分。

二、编写一个成绩分析程序。已知一个班有50名学生，每人参加6门考试。利用子程序方法求出每人的总分、平均分，并排出各科成绩的名次及总分名次。分数由DATA语句提供。

第十章　字符串函数及与机器码有关的语句

§1　字符串函数

在应用字符串时需要一些特有的函数。一般的电脑大都具有字符串函数。COMX—PC1电脑设置了ASC、CHR$、FVAL、LEN和MID$函数。

一、ASC函数

在电脑内存中，字符都是以一定的数码被储存的。最常用的数码是美国信息交换标准代码，即ASCII码。ASC函数可给出字符的ASCII码，ASCII函数的格式为：

ASC（字符串）

其功能是求出字符串第一个字符的ASCII码，以十进制形式输出。字符串可以赋给串变量。例如：

```
10  A$ = "ARITHMETIC"
20  B=ASC (A$)
30  PRINT  B
40  END
```

程序执行后的结果为：

```
65
```

字符A的ASCII码是65。注意ASC函数给出的值为浮点型。

二、LEN 函数

LEN 函数用来计算字符串字符的个数。它的一般格式

为：

LEN（字符串）

字符串可以是串变量。例如：

```
10  A$ = "BEIJING"
20  PRINT  LEN (A$)
30  END
```

结果为：

```
7
```

即 BEIJING 这个字符串有 7 个字符。

三、FVAL函数

FVAL 函数可将字符串表达式改变成算术表达式。FVAL 的格式为：

FVAL（字符串表达式）

FVAL 函数最多只能算出48个字符组成的表达式的值。例如：

```
10  A$ = "10+16"
20  PRINT  FVAL  (A$)
30  PRINT  FVAL  (A$ + "/2")
40  PRINT  FVAL  ("SQR (4)")
50  END
```

执行后的结果为：

```
26
18
 2
```

第20行是计算出10＋16的值。第30行是计算出10＋16／2的值。字符串用"＋"号联接相当于串联起来。第40行算出的

是4的平方根。

四、CHR$函数

CHR$ 函数是一种通用的输出字符函数。它可以通过字符（或图形）的ASCII码输出该字符（或图形）。一般用于PRINT语句中，有时也用在赋值语句中。CHR$的格式为：

CHR$（表达式，表达式，……）

表达式所决定的值是字符相应的ASCII码（十进制）。例如：

PRINT CHR$（65）

输出为A，因为A的ASCII码的十进制值为65。又如：

PRINT CHR$（67）

结果输出为C。如果打

PRINT CHR$（111，112，113）

屏幕上会显示出一架小飞机。而打

PRINT CHR$（111，112，112，113）

则会显示一架长机身的飞机。

在附录中可以查到各种字符的ASCII码值。

五、MID$ 函数

MID$函数的作用是从给定的字符串中选取某一部分组成子字符串。MID$函数的格式为：

MID$（字符串变量，表达式1，表达式2）

其中：字符串变量是给定的字符串的名字；
表达式1决定子字符串开始的字符位置；表达式2决定子字符串字符的个数，如果没有表达式2，就取至结尾。例如：

10 A$ = "GOOD"

```
20  B$=MID$ (A$, 3)
30  C$=MID$ (B$, 2)
40  PRINT  B$+C$
50  END
```

运行后的结果为：

```
ODD
```

又如：

```
10  A$= "EXPERIMENT"
20  PRINT  PID$ (A$, 7, 3)
30  END
```

执行后的结果为：

```
MEN
```

在某些BASIC版本中有LEFT$和RIGHT$函数。这两个函数的功能都能借助MID$函数来实现。

MID$ (A$, 1, N) 相当于 LEFT$ (A$, N)

MID$ (A$, (LEN (A$) - N + 1), LEN (A$)) 则相当于RIGHT$ (A$, N)。

例 10.1 数进制转换

```
10  DEFINT  Z: CPOS (0, 0): CLS
20  DIM  P (255)
30  PRINT "2 (B) -10 (D): <1>"
40  PRINT "10 (D) - 2 (B): <2>"
50  PRINT "16 (H) -10 (D): <3>"
60  PRINT "10 (D) -16 (H): <4>"
70  PRINT "END          <5>"
80  INPUT "SELECT 1, 2, 3, 4, 5" X
```

```
90  IFX<1  THEN  GOTO  80
95  IFX<5  THEN  GOTO  80
99  GOTO  100*X
100 INPUT "2 (B)  NUMBER" A$
110 L=LEN (A$)
120 FOR I=1  TO L
130     C$=MID$ (A$, I)
140     P (I) =FVAL (C$)
150 IF P (I) >1  THEN CPOS ( 0, 0): CLS
    : GOTO 30
160 NEXT I
170 N=2
180 GOSUB 1000
190 CPOS (0, 0) : CLS: GOTO 30
200 INPUT "10 (D) NUMBER" A$
210 A=FVAL (A$)
220 N=2
230 GOSUB 2000
240 GOSUB 3000
250 CPOS (0, 0) : CLS
260 GOTO 30
300 INPUT "16 (H) NUMBER" A$
310 GOSUB 4000
320 N=16
330 GOSUB 1000
340 CPOS (0, 0) : CLS
```

```
350 GOTO 30
400 INPUT "10 (D) NUMBER" A$
410 A=FVAL (A$)
420 N=16
430 GOSUB 2000
440 GOSUB 3000
450 CPOS (0, 0): CLS
460 GOTO 30
500 END
1000 S= 0: Q= 1
1010 FOR I= 1 TO L
1020 S=S+P (L-I+ 1) *Q
1030 Q=N*I
1040 NEXT I
1050 PRINT S
1060 RETURN
2000 I= 1
2010 Z=INT (A/N)
2020 IF Z<1 THEN P (I) =A: RETURN
2030 P (I) =A-Z*N
2040 A=Z
2050 I=I+ 1
2060 GOTO 2010
3000 FOR K=I TO 1 STEP - 1
```

```
3010 IF P(I)<10 THEN PRINT P(I);: GOTO
     3040
3020 Y=55+P (I)
3030 PRINT CHR$ (Y);
3040 NEXT I
3050 PRINT
3060 RETURN
4000 FOR I=1 TO L
4010 C$=MID$ (A$, I)
4020 IF ASC (C$) <58 THEN
     P (I) =FVAL (C$): GOTO 4040
4030 P (I) =ASC (C$) -55
4040 NEXT I
4050 RETURN
```

这个程序是利用字符串函数的功能进行数制间的变换。子程序1000行至1060行为N进制转为十进制。子程序2000行至2060行为十进制转为N进制。子程序3000行至3060行为输出N（N≠10）进制数。子程序4000行至4050行为十六进制数分离各位。整个程序基本是通过字符串函数来实现的，执行起来速度不快。但可以变换的数值范围很大，而且精确。编写此类程序的方法很多，这里就不一一列举了。

§2 与机器码有关的语句

在COMX BASIC中，可利用一些指令、语句、函数直

接与电脑内部发生关系。其中包括 EOP，EOD，MEM，PEEK，POKE，CALL，USR和SHAPE 等。

一、EOP和EOD指令

程序在电脑内存中所占有的区域叫程序区。数据占用的内存区域叫数据区。

EOP指令可以提供程序区中程序空间结尾处的十六进制地址。在程序输入时，利用EOP指令可以随时了解程序在内存中已经占用了多少单元。

EOD指令可提供数据区中数据所占空间结尾处的十六进制地址。在存取数据时，使用该指令可以帮助你了解数据量的大小。例如输入：

EOP：EOD

屏幕上会显示出

▌4411

▌411A

这表示程序结尾在4411（十六进制数），数据结尾在411A（十六进制数）。注意：EOD一定要在DIM语句之后使用才有效。

二、MEM函数

MEM函数用来检查电脑内存中还剩下多少存储单元，以十进制数给出。在程序输入完成后，检查内存占用情况，可打

PRINT MEM

这样就可确定此时还剩下多少存储单元。在设计程序时，要适当留出一定量的存储单元。

三、PEEK函数

PEEK函数可列出确定地址的存储单元中的内容，它的值为十进制数。例如：

```
10  A=PEEK (16842)
20  IF A=8 THEN PRINT"NTSC MACHINE"
30  IF A=9 THEN PRINT"PAL MACHINE"
40  END
```

程序执行后，首先将16842地址内存中的内容以十进制数赋给变量A。如果A的值是8则打印"NTSC MACHINE"，说明电脑以NTSC制输出；如果A的值为9则输出"PAL MACHINE"，说明电脑以PAL制输出。

四、DEFUS 语句

DEFUS 语句用来确定用户区起始地址位置。它通常的格式为：

DEFUS 表达式

表达式的值为用户程序开始的地址。不使用该语句时，用户程序将以4400（十六进制）开始。若使用 DEFUS 语句，其表达式值应大于4400。在4400至确定起始地址之间可以写入机器语言子程序。

起始地址位置都是按整页处理的。每256(十进制）个存储单元为一页。例如：

DEFUS ▮4450

程序开始地址实际为44FF（十六进制）。

如果确定起始地址小于4400（十六进制），会产生错误。因为此时进入了系统区，电脑不会接受。值得注意的是，一旦使用了DFEUS语句，程序的首地址就确定了。如要改变首地址需要再使用 DEFUS 语句。

五、POKE语句

POKE语句是BASIC语言中一个特有的语句。它的功能是在BASIC控制下，直接利用机器语言与电脑发生联系。POKE 语句的格式为：

行号　POKE（表达式 1，表达式 2）

POKE的功能是将表达式 2 确定的值写到表达式 1 确定的地址中去。即：

表达式 1　表示地址码

表达式 2　表示机器码

例如：

POKE（▮4400，255）

是将255这个数写入到4400（十六制）地址中去。这个语句相当

POKE（▮4400　#FF）

不要随便使用 POKE 语句。需用时也要当心。用得不好会破坏程序的正常执行。

六、CALL 语句和 USR 函数

CALL语句是调用机器语言子程序的指令。CALL 的格式为：

行号　CALL（表达式）

式中表达式的值为机器语言子程序开始的地址。一般采用十六进制数。例如：

CALL（▮4400）

表示调用以4400（十六进制）地址开始的机器语言子程序。

USR 是一个函数，可以做为表达式的一部分。一般是在机器语言子程序中调用这个函数。

关于机器语言子程序的编写可见书后的附录。

七、定义字符函数

在电脑中的 ROM 中存有一些固定的字符和图形，通过 CHR $ 函数可以输出这些字符、图形。若想改变这些字符的形状，可以使用定义字符图形 SHAPE。这样就可以使电脑屏幕输出的图形灵活机动，丰富多采。SHAPE 函数的格式为：

行号　SHAPE（X，“Y”）

式中X为字符的ASCII码，Y为十八位十六位进制数。

屏幕上显示的字符由 6 × 9 个点组成，每个字符占内存的 9 个字节。每个字节的前两位用来定义字符颜色，其后六位定义显示点。凡需显示的点用 1 表示，不显示的点用 0 表示。每个字节为二位十六进制数， 9 个字节共十八位十六进制数。前二位（左二位）定义的颜色见表10.1

表 10.1　　字符颜色定义表

左二位	屏幕输出色	键盘输入色
0　0	黑色	红色
0　1	蓝色	粉红色
1　0	绿色	黄色
1　1	青色	白色

假如要将A键定义成“中”字，可用如下指令：

SHAPE（65，“48487E6A7E48484800”）65为A键的 ASCII 码，后面的十六进制数定义了“中”字，见图10.1。现在只要按A键，屏幕上就会显示出“中”字。

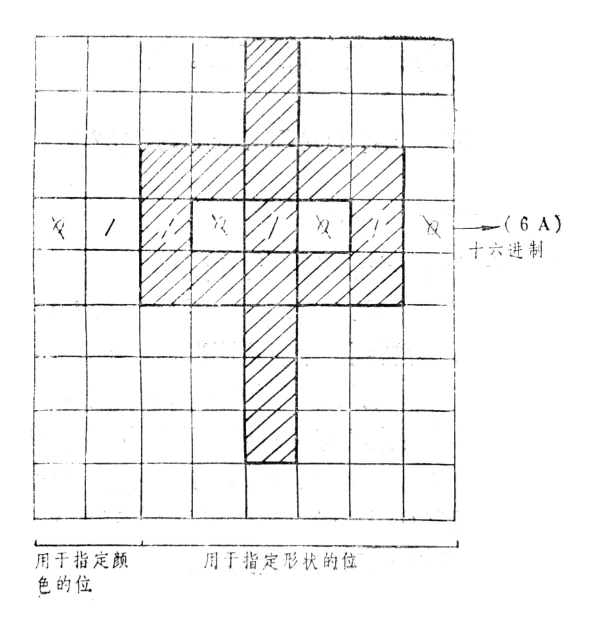


图 10.1 定义"中"字

利用定义字符函数可以构成汉字库。通常 6 × 9 个点定义的汉字不太好看，可以用四个字符定义一个汉字。例如：用四个字符定义一个"电"字。程序如下：

```
10  SHAPE (156, "0081819F9191919F91")
20  SHAPE (157, "000000BC848484BC84")
30  SHAPE (158, "91919F818181810000")
40  SHAPE (159, "8484BC008181BF0000")
50  CPOS (10, 12) : PRINT CHR$(156, 157)
60  CPOS (11, 12) : PRINT CHR$(158, 159)
70  END
```

程序执行后即可在屏幕上显示一个"电"字。

除了定义字符外，还可以定义图形。例如要想画一个苹果可借助如下语句。

SHAPE (153, “8F8D8C5C7E7E5C00”)

打

PRINT CHR$ (153)

就可以在屏幕上看到一个苹果图形。再如：

SHAPE (119, “34848CBF9E9C94A400”)

按 SHIFT—W键就会看到一个星星图形。

COMX—PC1电脑全屏幕共有240×216个点。要想输出一个理想的图形，首先要考虑总体结构应由哪些单元构成，再根据需要定义若干字符、图形，最后进行整体拼凑。此外，当定义大量字符图形时，可以用数据语句 DATA 建立数据区，采用循环方法画出整幅图形。

在定义字符、图形后，要想恢复机内定义字符只需打

COLOR (12)

即可完全复原。若只想将部分定义的字符恢复为原机内字符，必须再行定义原字符。本书附录中介绍了若干机内字符的定义码。

§3 练 习 题

一、用四个字符定义一个汉字，一共定义十个汉字。

二、画一个充满全屏幕的几何图形。

附 录 A

COMX—PC 1 电脑外存储设备的使用

电脑一旦断电，其中的所有程序都会被清除掉。利用外存储设备可以使所有有用的信息记录在磁带上。下面介绍存取信息的方法和需要注意的有关事项。

§ 1 将程序存到磁带上

一般情况下，录音机要先接好，以免临时接入时由于电源电压的影响破坏程序。

在一个 BASIC 程序（也可以是不完整的程序）调试完成以后，按下列步骤操作。

1．将录音磁带卡入录音机内。

2．记下录音机计数器的数字。

3．在电脑键盘上打 PSAVE。

4．置录音机于录音的位置。

5．按 CR 键。此时光标消失，可听到高频叫声，这是程序头，接着出现噪音。

6．待程序录完后，屏幕上重新出现

READY

：♦

的信息时，按下停止键使录音停止。记录下计数器的数字。

7．建立程序记录。在纸上记下程序名，首数字及尾数

字、日期等。

至此，存程序的全部工作已完成，这时程序仍然在电脑中，打LIST即可显示。

如果将电脑的电源关闭再打开，然后打入LIST，则无程序显示。不过不要紧，程序已经存在磁带上了。

注意：

1．如果在打 PSAVE 之后录音机发生任何故障都不要关掉电脑。因为录音机发生故障不会影响电脑存程序的工作，待程序“存”完后，屏幕上仍然会显示出如下信息：

READY

：◆

此时程序并没有存入磁带，排除故障后可以再重新进行存程序的工作。

2．在存程序过程中，声音也是有一定规律的。尖叫声是程序头。噪音表示转送的页（每页为256字节），每页之间有短促叫声，这叫页头。最后结束的标志是尖叫声之后接低音。如果熟悉这些声音的变化，就可以知道程序大概送存到什么地方了。

3．不是所有的录音机都能听到相应的声音的，如果所用的录音机听不到这些声音也不要担心，只要观察屏幕就可以了。

§2　将磁带中的程序调回电脑中

将磁带上记录的程序调回到电脑中是需要细心操作的。这除了能调回你录存的程序外，还可能直接调取别人编制的

程序，特别是一些游戏程序。因为较复杂的程序一般都要经过这项操作才能提供给用户使用。

调回程序给电脑按下列步骤操作。

1．装好录有程序的磁带并找到程序开始的位置。可直接听，也可按计数器上的数字找。

2．将录音机的音量调大一些，音调调高一点。

3．考虑并检查一下电脑中原有的程序或数据是否已不再需要，因为一旦取程序工作开始，电脑中原有的信息均会被清除。

4．在键盘上打PLOAD，然后按录音机上的PLAY钮，当听到尖叫声后，再按CR键。

5．如果没有出现什么故障，待调程序工作完成后，屏幕上会出现。

READY

：◆

信息。关掉录音机。（不忙于关录音机电源，因为这样可能会破坏程序。）

此时，程序已调回电脑内存中了。可以打LIST检查一下。

注意：

1．调回程序的过程中可能会出现

ERR 57

的信息，这说明调程序的工作失败，需要重新进行。也可能会出现

READ TAPE ERR

的信息，这说明磁带上的信息在调回过程中发生了一般性错

误。如果出现三个以上这种错误，调取程序工作必须重新进行。

2．一旦在调取程序的过程中遇到各种错误信息（上述两种为典型错误信息），在重新工作时必须做如下处理：

（1）仔细检查输出、输入接口、插头及连线。

（2）仔细调节录音机的音量和音调，以达到最佳效果。

（3）如果录音机确有问题就调换另一台。

§3　有关数据存取的命令

COMX—PC1电脑提供了专门用来存取数据的指令。它们是

DSAVE　（存数据）

DLOAD　（取数据）

存数据就是将内存中数据区的内容存到磁带上。取数据就是将先前由DSAVE命令存入磁带上的数据调回到电脑中。取数据工作执行后会将数据放置在程序空间后面并冲掉原有的数据。

DSAVE的使用与PSAVE的使用相同；DLOAD的使用与PLOAD的使用也相同。操作步骤可参照前面所介绍的办法。

数据存取多用于程序中，特别是建立数据文件时更需要这种方法。

注意：

1．在内存中同时包含程序和数据时，对程序的任何编

辑都会把数据抹去。因此，在对程序进行编辑之前，一定要考虑好数据是否要存的问题。

2．字符串是放在数组空间末尾，因而，在定义字符串之前一定要先将数组说明，否则会抹去字符串。

§4 使用外存储设备的注意事项

1．使用高质量的磁带存信息。

2．使用尽可能短的磁带。

3．保持连线两头接触良好。

4．不要将录有程序的磁带放在强磁场附近。

5．尽量保证录音机的工作电压正常。

6．使用同一录音机做存、取工作。

7．经常保持磁头清洁。

8．定期给磁带录音机消磁。

附　录　B

COMX—35P打印机及功能扩展板的使用

使用COMX—35P打印机及功能扩展板首先应具备全系统（录音机不用时可不接入）。一定要在开始时就将各部分装置连接好，不要中途插入扩展板。打印机开关也可先打开。

§1　COMX—35P打印机操作命令

打印机的所有操作都是通过功能扩展板内的ROM控制的。统一采用CALL命令。

执行打印有若干个进入点地址，下面分别介绍。

一、打印机输出

采用

CALL（▮C060）

或

CALL（▮C040，1）

或

CALL（▮C070）

命令。这种操作是关闭电脑显示屏幕，结果以文本输出。也就是说结果以打印机按标准打印格式打印出来。此种方式也可以直接从键盘输入打印。

CALL（▮C060）及CALL（▮C040，1）禁止使用CTRL—和SHIFT—字符打印，而CALL（▮C070）不限。

二、关闭打印机

采用

CALL（▮C050）

或

CALL（▮C040，0）

命令。这个操作是关闭打印机并且使结果由屏幕输出。

三、打印机、显示屏同时输出

采用

CALL（▮C080）

或

CALL（▮C040，3）

或

CALL（▮C090）

命令。这种操作可给出标准打印格式的文本输出，还可以打印键盘输入的符号

CALL（▮C080）及CALL（▮C040，3）命令不能完成CTRL—和SHIFT—字符输出。CALL（▮C090）可以完成任何字符输出。

四、取消键盘输入字符打印

采用

CALL（▮C0A0）

或

CALL（▮C040，5）

命令。这种方式使打印机停止打印由键盘输入的字符，而屏幕上仍可以输出键盘输入的字符。

五、紧凑行距打印

为了使文本输出节省纸张，打印紧凑行距方式采用

CALL（▮C0D0）

或

CALL（▮C040，8）

命令。这种方式使行与行之间的距离很小。但图形输出只能打印出机内定义的图形、字符，用户自定义字符不能打印。

六、自检打印

采用

CALL（▮C0E0）

或

CALL（▮C040，9）

命令。这种方式可以在打印机上打印出一系列标准字符，但屏幕上无输出。

七、屏幕转储方式

屏幕上显示的信息可以转储，并且可以按用户的要求在打印机上输出。这种操作方式有三个命令：

1．屏幕图象形成

采用

CALL（▮C0F0，1）

命令。这个操作将内存BDFF地址以下至BA00地址之间的内容转移到系统堆栈内。该区间保留1K的存储单元用来形成屏幕图象。此命令执行后，屏幕图象就保留在BA00至BDFF的内存中。这个命令常用于应用程序中。

2．屏幕图象转储

采用

CALL（▮C0F0，2，X）

命令。这个操作是使打印机输出打印图象。其中参数X是行的数量，全屏幕共24行，如果想让打印结果充满全屏幕则X为24。该命令用于图象形成命令之后。

3．取消屏幕图象

采用

CALL（▮C0F0，3）

命令。这个操作是将系统堆栈的内容转回到原来的内存地址中去。它用于图象转储命令之后。

九、显示入口地址命令

如果想在屏幕上显示入口地址可采用

CALL（▮C040，11）

命令。该命令执行后，屏幕上应为：

C050－PRINTER OFF　　(LF)
C060－TEXT　　　　　　(SP)
C080－TEXT　　　　　　(TV，SP)
C0A0－KEYBOARD OFF
C0C0－MEM DUMP
C0D0－TEXT　　　　　　(NO SP)
C0E0－SELE TEST
C106－PRINTER OFF　　(LF)
C10C－MEM MOVE
C10F－MEM MODIFY

在文本与图形方式互相转换时，应该执行 CALL

(▌C050)或CALL(▌C106)命令，然后再转向图形方式；或执行CALL(▌C0F0，3)命令后再转向文本方式。不这样处理，就会产生一些错误，致使打印中断。

§2 打印机使用时可能出现的错误信息

在使用打印机时会发生一些常见错误，这时屏幕上会给出错误信息。通常有以下三种：

一、准备不当

信息为：

THERMAL PRINTER NOT READY

这说明打印机没准备好，可能是电源未接通或没有打印纸。

二、命令错误

信息为：

WRONG MODE SELECT CODE

这说明打印机不能接受输入的命令。

三、超限错误

信息为：

ENERGIZING TIME EXCEEDS 1.6ms,
THIS AFFECTS THE OPERATION OF THE PRINTER

这说明激发时间超过1.6ms，需停止打印。

出现错误信息后，需重新进行操作。但这并不影响前面已经进行过的工作。

§ 3 例　　题

例 B.1　在打印机上印出全部程序清单。

打入

CALL（■C060）：LIST：CALL（■C050）

按CR键后即可实现。

例 B.2　在打印机上印出10至400行的程序。

打入

CALL（■C070）：LIST 10，400：CALL（■C050）

按CR键后即可实现。

例 B.3　打印出ASC码为1至128所对应的字符。

程序如下：

```
10  CALL (■C070)
20  FOR  I = 1  TO  128
30  PRINT  I, CHR$ (I)
40  NEXT  I
50  CALL (■C050)
60  END
```

程序输入后打RUN，按CR键即可。

例 B.4 打印出以下图形

```
* * * * * * * * * * * * * * * * * * * * * * * *
*                                             *
*   * * * * * * * * * * * * * * * * * * * *   *
*   *                                     *   *
*   *   * * * * * * * * * * * * * * * *   *   *
*   *   *                             *   *   *
*   *   *   * * * * * * * * * * * *   *   *   *
*   *   *   *                     *   *   *   *
*   *   *   *   * * * * * * * *   *   *   *   *
*   *   *   *   *             *   *   *   *   *
*   *   *   *   * * * * * * * *   *   *   *   *
*   *   *   *   *             *   *   *   *   *
*   *   *   *   * * * * * * * *   *   *   *   *
*   *   *   *                     *   *   *   *
*   *   *   * * * * * * * * * * * *   *   *   *
*   *   *                             *   *   *
*   *   * * * * * * * * * * * * * * * *   *   *
*   *                                     *   *
*   * * * * * * * * * * * * * * * * * * * *   *
*                                             *
* * * * * * * * * * * * * * * * * * * * * * * *
```

程序如下：

```
100 CPOS (0, 0) : CLS
110 CALL (▮C0F0, 1)
120 H= 0 : X 1 =40: Y 1 =21
130 N= 1 : H=H+ 2 : X1=X1- 2 : Y1=Y1- 2
140 X=H: Y=H
150 IF N= 0 THEN GOTO 200
160 FOR K=Y TO Y 1
170 CPOS (K, X) : PRINT "*"
180 NEXT K: Y=Y 1
```

```
190 IF N=0  THEN GOTO 130
200 CPOS (Y, X) : X=X+1 : PRINT "*"
210 IF X<X1  THEN GOTO 200
220 IF N=0  THEN GOTO 160
230 IF Y1<12  THEN GOTO 260
240 N=0
250 GOTO 140
260 CALL (■C0F0, 2, 24)
270 CALL (■C0F0, 3)
280 END
```

在打印过程中，按CTRL－P可以中断打印。

§4　功能扩展板的使用

打印机需要扩展板才能使用，这在前面已经介绍过了。这一节介绍如何利用功能扩展板做下述工作。

一、查看电脑内存中的内容

通过功能扩展板可以直接查看电脑内存中的内容，其命令为

CALL (■C109, X, Y)

其中参数 X 为要查看内容的首地址，Y 为要查看内容的尾地址，地址用十六进制数表示。

该命令用于串行输出方式时，可以选择输出装置（屏幕或打印机）。

例如，要在屏幕上查看地址0600至0700之间存储器中的内容。先打入

CALL (▮C109, ▮0600, ▮0700)

这时屏幕上显示出

OUTPUT TO SCREEN(S) OR PRINTER (P)

: ♦

(屏幕输出打S，打印机输出打P）我们想让结果在屏幕上显示，因而需按S键，然后按 CR 键。此时屏幕上显示出内存中的内容：

```
0600 89 8B 8A 86 0A 00 00 00 . . . . . . . .
0608 00 00 00 00 00 00 FE D4 . . . . . . . .
0610 D4 D4 D4 D4 D4 00 00 FE  . . . . . . . .
0618 E2 D0 C8 D0 E2 FE 00 00  . . . . . . . .
0620 1E 20 20 20 20 20 1E 00 . . . . . . . .
0628 00 FF E1 E1 E1 E1 E1 E1  . . . . . . . .
0630 E1 FF C0 DE DE DE DE DE  . . . . . . . .
0638 DE DE C0 FF 00 00 00 00 . . . . . . . .
0640 00 00 00 00 C1 C1 C1 C1 . . . . . . . .
0648 C1 C1 C1 C1 C1 00 00 00 . . . . . . . .
0650 00 00 00 00 00 FF 1C 08 . . . . . . . .
0658 08 08 08 08 1C 00 00 0E . . . . . . . .
0660 04 04 04 04 24 18 00 00 . . . . $ . . .
0668 22 24 28 30 28 24 22 00  "$ ( 0 ( $ " .
0670 00 E0 E0 E0 E0 E0 E0 E0  . . . . . . . .
0678 E0 E0 22 36 2A 2A 22 22 . .  " 6 * * " "
0680 22 00 00 C8 C8 C8 C8 FE  ". . . . . . .
0688 DC C8 00 00 00 C8 CC FE  . . . . . . . .
0690 CC C8 00 00 00 FF E0 F0  . . . . . . . .
```

```
0698 E0 E0 E0 E0 E0 E0 FF C1 . . . . . . . . .
06A0 C1 C1 C1 C1 C1 C1 C1 3C . . . . . . . . <
06A8 22 22 3C 23 24 22 00 00 " " < (& " . .
06B0 1C 22 20 1C 02 22 1C 00 . " . . " . . .
06B8 00 E0 E0 E0 E0 E0 E0 E0 . . . . . . . .
06C0 E0 FF CC DE FF FF FF FF .. .. .. ..
06C8 FF DE CC CC CC CC CC CC .. .. .. ..
06D0 CC CC CC CC 00 00 00 FF . . . . . . . .
06D8 FF FF 00 00 00 CC CC CC . . . . . . . .
06F8 D8 FE D8 C8 00 00 00 C1 . . . . . . . .
06F0 C1 C1 C1 C1 C1 C1 C1 FF . . . . . . . .
0700 00
```

二、拷贝256字节的存储器内容

如果想将某段地址内的内容拷贝到一段新地址中，可以利用拷贝命令。其格式为：

CALL (▮C10C, X, Y)

式中参数X是原内存的末地址，参数Y是新内存的末地址。这个命令不必选择输出装置。

例：将700至800地址中的内容拷贝到5000至5100地址中去。

打入

CALL (▮C10C, ▮0800, ▮5100)

屏幕上出现：

```
SOURCE = 0800   DEST = 5100   COPY (Y/N)? Y
SOURCE = 0700   DEST = 5000   COPY (Y/N)? Y
SOURCE = 0600   DEST = 4F00   COPY (Y/N)? N
```

并显示出

READY

:♦

这时，拷贝工作已经完成。可查看一下0700至0800中的内容与5000至5100中的内容是否一致。这里列出5000至5100之间内存中的内容

```
CALL (▮C109, ▮5000, ▮5100)
OUTPUT TO SCREEN (S) OR PRINTER (P)
:S
5000 00 00 00 00 00 00 00 00 . . . . . . . .
5008 00 00 00 00 00 00 00 00 . . . . . . . .
5010 00 00 00 00 00 00 00 00 . . . . . . . .
5018 00 00 00 00 00 00 00 00 . . . . . . . .
5020 00 00 00 00 00 00 00 00 . . . . . . . .
5028 00 00 00 00 00 00 C8 C8 . . . . . . . .
5030 C8 C8 C8 O8 C8 00 00 D4 . . . . . . . .
5038 D4 00 00 00 00 00 00 00 . . . . . . . .
5040 D4 FE D4 D4 FE D4 D4 00 . . . . . . . .
5048 00 C8 DE E8 DC CA FC C8 .... ....
5050 00 00 F2 F2 C4 C8 D0 E6 . . . . . . . .
5058 E6 00 00 00 D0 E8 E8 D0 . . . . . . . .
5060 EA E4 DA 00 C8 C8 C0 C0 . . . . . . . .
5068 C0 C0 C0 00 00 C2 C4 C8 . . . . . . . .
5070 C8 C8 C4 C2 00 00 E0 D0 . . . . . . . .
5078 C8 C8 C8 D0 E0 00 00 C8 . . . . . . . ..
5080 EA DC C8 DC EA C8 00 00 . . . . . . . .
```

```
5088 00 C8 C8 FE C8 C8 00 00 . . . . . . . . . .
5090 00 C0 00 C0 00 C0 C8 C8 . . . . . . . . . .
5098 D0 00 00 00 00 FE 00 00 . . . . . . . . . .
50A0 00 00 00 C0 C0 C0 C0 C0 . . . . . . . . . .
50A8 C0 C8 C0 00 00 C2 C4 C8 . . . . . . . . . .
50B0 D0 E0 00 00 00 DC E2 E6 . . . . . . . . . .
50B8 EA F2 E2 DC 00 00 C8 C8 . . . . . . . . . .
50C0 C8 C8 C8 C8 DC 00 00 DC . . . . . . . . . .
50C8 E2 E2 CC D0 E0 FE 00 00 . . . . . . . . . .
50D0 DC E2 C2 CC C2 E2 DC 00 .... .....
50D8 00 C4 CC C4 E4 FE C4 C4 . . . . . . . . . .
50E0 00 00 FE F0 FC C2 C2 E2 . . . . . . . . . .
50E8 DC 00 00 CC D0 E0 FC E2 . . . . . . . . . .
50F0 E2 DC 00 00 FE C2 C4 C8 . . . . . . . . . .
50F8 D0 D0 D0 00 00 DC E2 E2 . . . . . . . . . .
5100 DC
```

注意：

拷贝命令中的Y参数不能是ROM的地址码因为ROM 是不能写入的。

例如：

```
CALL (▮C10C, ▮0900, ▮C003)
SOURCE=0900 DEST=C003 COPY (Y/N) ? Y
FAIL  AT  C003  WR=00  DR=FF
FAIL  AT  C002  WR=C8  DR=FF
FAIL  AT  C001  WR=C8  DR=02
. . .
```

由于地址C003、C002等内存为ROM区，因而不能写入。

三、改变内存储器中的内容

可以利用功能扩展板改变RAM中的内容。其命令为：

CALL (■C10F)

这个命令可以改变RAM中任一字节内容，但不能改变ROM中的内容。

例如：打入

CALL (■C10F)

屏幕上出现

MEM ADDR: ◆

此时可以输入RAM的地址数，打入

4900

屏幕上会出现如下信息：

```
4900  FF  50  (改变值为50)
4901  FF  51  (改变值为51)
4902  FF  52
4903  FF  53
4904  FF  54
4905  FF  55
4906  FF  56
4907  FF  57
4908  FF  58
4909  FF  59
490A  FF  5A
490B  FF  5B
490C  FF  5C
```

```
490D  FF  5D
490E  FF  5E
490F  FF  5F
4910  FF        (按空格棒可保留原内容)
4911  FF
4912  FF
4913  FF
4914  FF
4915  FF
4916  FF  ·     (打“·”则向回列出地址)
4915  FF  D1
4914  FF  D2
4913  FF  D3
4912  FF  D4
4911  FF  D5
4910  FF  D6
490F  5F        (按空格棒)
490E  5E        (这些地址中的内容已经改变了)
490D  5D
490C  5C
……
```

如果要结束这项工作可以按ESC键。

现在查看一下4900至4910地址中的内容。

打入

CALL (▮C109, ▮4900, ▮491F)

屏幕上出现

OUTPUT TO SCREEN(S) OR PRINTER(P):

打入S，屏幕上显示出：

4900 50 51 52 53 54 55 56 57 PQRS TUVW

4908 58 59 5A 5B 5C 5D 5E 5F XYZ〔 \〕↑—

4910 FF

ROM中的内容是不能改变的。例如，打入

CALL (▮C10F)

屏幕上显示出

MEM ADDR:

打入0900，接着出现

0900 00

打入40，显示为

0900 00 40 CHANGE FALL: M=00

这表明ROM中的内容是改变不了的。

利用这种改变内存内容的输入方式可以编写机器语言子程序。机器语言子程序比BASIC语言程序执行速度要快很多，一般在游戏程序中经常用到。学习机器语言要求掌握1802A的指令系统，这些在附录I中有简单介绍。

附 录 C

COMX—PC 1 电脑显示器的制式及电脑的调整

§1 显示器的制式

电脑显示器选用的电视接收机一般有两种制式：PAL制和NTSC制。这两种制式是不相同的。

一、图形字符

在PAL制中每个字符由 9 条水平线构成，而 NTSC 制中每个字符由 8 条水平线构成。

在NTSC制电视系统使用SHAPE 指令时，Y 参数的最后两位数字是无效的。

二、计时装置

PAL制中的内部计时器比率是每秒50次，而 NTSC 制的是每秒60次。因而，在使用TIME指令时需要考虑参数值的选取。

三、测试图

在COMX—PC 1 电脑刚刚开机时，可以从测试图上看出电脑处于何种制式输出。两条水平线在PAL制 或 NTSC 制中会显示出不同的颜色序列。

————————————……彩色杠

C O M X

————————————……彩色杠

当彩色杠的颜色为绿色或蓝色时，说明此时电脑输出为PAL制；当彩色杠的颜色为青色或黄色时，说明此时电脑输出为NTSC制。根据彩色杠的颜色判断出电脑的制式，再根据制式来选择电视机。

§2 电脑的简单调整

一般情况下，无需对电脑进行调整。确有必要可以通过电脑底边的三个小孔做些调整。

这三个小孔分别标有MIX，SYNC和VIDO控制，其功能和调整步骤简述如下。

一．MIX

MIX用于调整颜色信号的频率，以使亮度信号与色度信号之间的干扰最小。

如果该控制调整不够理想，在显示屏上的字符边缘会出现一些移动的或固定的彩色点。

调整MIX可按以下步骤进行。

1．打入以下程序：

```
10  SHAPE (#40, "FF FF FF FF FF FF FF
    FF FF")
20  PRINT "1 11 11 11 11 11 11
    11 11 11 11 11 11 11 11 11
```

```
    11 11 11 11 11 11 11 11 11
    11 1"
30  FOR A=1  TO  22
40  PRINT CHR$ (107, 108, 64, 237, 235,
    236, 192);
50      FOR I=1  TO  8
60      READ X
70        FOR  J=1  TO  4
80        PRINT CHR$ (X);
90        NEXT  J
100      NEXT  I
110      PRINT "              ";
120   RESTORE
130   NEXT  A
140   B=KEY
150   IF  B< > 69  THEN GOTO  140
160   END
```

执行此程序。

2.调 MIX 小孔中的螺丝。观察数字“1”的边缘和彩色竖线之间的地方，缓慢地仔细转动螺丝，直至依附在字符上的彩色点不出现为止。

有些电脑显示屏的彩色在调节时会全部变成黑白色，这时只要反方向转动螺丝，彩色会重新出现。

二、SYNC

SYNC孔用来调整同步脉冲的振幅，调整时要十分小心，一般应有示波器配合。

三、VIDEO

VIDEO 孔用来调整送给RF调制器的图象信号的振幅，振幅低使图象模糊，振幅高使黄色显示为白色。一般情况下不要调节这个控制孔。

附 录 D

十进制、十六进制、二进制换算表

十进制	十六进制	二进制
0	0	0
1	1	1
2	2	10
3	3	11
4	4	100
5	5	101
6	6	110
7	7	111
8	8	1000
9	9	1001
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111

附 录 E

COMX—PC 1 电脑内存分配表

首地址	末地址	功 能	容 量
F800	FF	显示屏 RAM	2K
F400	F7FF	字 符 RAM	1K
F000	F3FF	保 留	1K
E400	EFFF	保 留	4K
C000	DFFF	存储库	8K
BE00	BFFF	保 留	
4000	BDFF	堆栈、系统 RAM BASIC程序区， 机器语言子程序区	32K
0000	3FFF	系 统 ROM	16K

说明

0000—3FFF

内部系统ROM。包含（1）BASIC解释程序。（2）电视和键盘操作系统。（3）磁带操作系统

4000—43FF

系统参数存储区。

4400—BDFF

BASIC程序和数据存储区。系统堆栈的顶部在BDFF。

BE00—BFFF

8K字节的8个存储库。用于系统的扩展。对于某个区引进读、写操作之前程序必须先选择所要的存储库。

E000—EFFF

保留。用于存储库之间的通讯；不能适应于一个8K字节存储器的应用。

F000—F3FF

保留。

F400—F7FF

字符定义的RAM区。这个区保持128个字符的点模式。字符可由BASIC中的SHAPE指令改变。这个区只能由转到CDP1869 IC的“字符存储存取”的状态来访问。

F800—FFFF

显示屏页面存储器。每一个字节保持屏上一个显示字符位的字符代码。在写入之前，F800代表屏上左上角的字符位置，F801紧挨F800右边的一个位置，F803代表F802右边的一位置……等等。EBBF代表屏上右下角位置。硬件写入是通过改变在CDP1869 IC里面的家庭地址寄存器完成的。该区是只写存储区，而且只能在没有显示的时间访问，程序可以等到没有显示的时候再直接写入这个区。

附 录 F

键盘字符及机内字符的ASCII码

ASCII码		按 键	图形	输出颜色
十进制	十六进制			
0	0			
1	1	CTRL—A	╥	青色
2	2	CTRL—B	Σ	青色
3	3			
4	4	CTRL—D	□	青色
5	5	CTRL—E	■	青色
6	6	CTRL—F		
7	7	CTRL—G	|	青色
8	8	CTRL—H	—	青色
9	9			
10	A	LF		
11	B			
12	C	CTRL—L	|	青色
13	D	CR		
14	E	CTRL—N	↓	青色
15	F	CTRL—O	→	青色
16	10	CTRL—P	┌	青色

ASCII码		按　键	图形	输出颜色
十进制	十六进制			
17	11	CTRL—Q	┓	青色
18	12			
19	13			
20	14	CTRL—T	┗	青色
21	15	CTRL—U	♦	青色
22	16	CTRL—V	❘	青色
23	17	CTRL—W	━	青色
24	18	CTRL—X	╋	青色
25	19	CTRL—Y	◀	青色
26	1A	CTRL—Z	┛	青色
27	1B			
28	1C			
29	1D			
30	1E			
31	1F			
32	20	空格棒		
33	21	!	!	青色
34	22	"	"	青色
35	23	#	#	青色
36	24	$	$	青色
37	25	%	%	青色
38	26	&	&	青色
39	27	'	'	青色
40	28	(	(	青色

ASCII码		按 键	图形	输出颜色
十进制	十六进制			
41	29	)	)	青色
42	2A	*	*	青色
43	2B	+	+	青色
44	2C	,	,	青色
45	2D	—	—	青色
46	2E	.	.	青色
47	2F	/	/	青色
48	30	0	0	青色
49	31	1	1	青色
50	32	2	2	青色
51	33	3	3	青色
52	34	4	4	青色
53	35	5	5	青色
54	36	6	6	青色
55	37	7	7	青色
56	38	8	8	青色
57	39	9	9	青色
58	3A	:	:	青色
59	3B	;	;	青色
60	3C	<	<	青色
61	3D	=	=	青色
62	3E	>	>	青色
63	3F	?	?	青色
64	40	█	█	青色

ASCII码		按　键	图形	输出颜色
十进制	十六进制			
65	41	A	A	青色
66	42	B	B	青色
67	43	C	C	青色
68	44	D	D	青色
69	45	E	E	青色
70	46	F	F	青色
71	47	G	G	青色
72	48	H	H	青色
73	49	I	I	青色
74	4A	J	J	青色
75	4B	K	K	青色
76	4C	L	L	青色
77	4D	M	M	青色
78	4E	N	N	青色
79	4F	O	O	青色
80	50	P	P	青色
81	51	Q	Q	青色
82	52	R	R	青色
83	53	S	S	青色
84	54	T	T	青色
85	55	U	U	青色
86	56	V	V	青色
87	57	W	W	青色
88	58	X	X	青色

ASCII码		按　键	图形	输出颜色
十进制	十六进制			
89	59	Y	Y	青色
90	5A	Z	Z	青色
91	5B	[	[	青色
92	5C	\	\	青色
93	5D	]	]	青色
94	5E	∧	↑	青色
95	5F	—	—	青色
96	60		0	绿色
97	61	SHIFT—A	1	绿色
98	62	SHIFT—B	2	绿色
99	63	SHIFT—C	3	绿色
100	64	SHIFT—D	4	绿色
101	65	SHIFT—E	5	绿色
102	66	SHIFT—F	6	绿色
103	67	SHIFT—G	7	绿色
104	68	SHIFT—H	8	绿色
105	69	SHIFT—I	9	绿色
106	6A	SHIFT—J	0	绿色
107	6B	SHIFT—K	■	绿色
108	6C	SHIFT—L	■	蓝色
109	6D	SHIFT—M	■	黑色
110	6E	SHIFT—N	[illegible]	青色
111	6F	SHIFT—O	[illegible]	青色
112	70	SHIFT—P	[illegible]	青色

ASCII码		按 键	图形	输出颜色
十进制	十六进制			
113	71	SHIFT—Q	[illegible]	青色
114	72	SHIFT—R	♪	青色
115	73	SHIFT—S	⇑	青色
116	74	SHIFT—T	C	青色
117	75	SHIFT—U	O	青色
118	76	SHIFT—V	M	青色
119	77	SHIFT—W	X	青色
120	78	SHIFT—X	–	绿色
121	79	SHIFT—Y	C	青色
122	7A	SHIFT—Z	[illegible]	绿色
123	7B		M	蓝色
124	7C			
125	7D		–	绿色
126	7E		©	绿色
127	7F		◆	蓝色
144	90		┏	白色
145	91		┓	白色
146	92			
147	93			
148	94		L	白色
149	95		●	白色
150	96		|	白色
151	97		–	白色
152	98		+	白色

ASCII码		按　键	图形	输出颜色
十进制	十六进制			
153	99		←	白色
154	9A		┘	白色
155	9B			
156	9C			
157	9D			
158	9E			
159	9F			
160	A0			
161	A1		!	白色
162	A2		"	白色
163	A3		#	白色
164	A4		$	白色
165	A5		%	白色
166	A6		&	白色
167	A7		'	白色
168	A8		(	白色
169	A9		)	白色
170	AA		*	白色
171	AB		+	白色
172	AC		,	白色
173	AD		—	白色
174	AE		.	白色
175	AF		/	白色
176	B0		0	白色

ASCII码		按　键	图形	输出颜色
十进制	十六进制			
177	B1	⋮	1	白色
178	B2	⋮	2	白色
179	B3		3	白色
180	B4		4	白色
181	B5		5	白色
182	B6		6	白色
183	B7		7	白色
184	B8		8	白色
185	B9		9	白色
186	BA		:	白色
187	BB		;	白色
188	BC		<	白色
189	BD		=	白色
190	BE		>	白色
191	BF		?	白色
192	C0		▬	白色
193	C1		A	白色
194	C2		B	白色
195	C3		C	白色
196	C4		D	白色
197	C5		E	白色
198	C6		F	白色
199	C7		G	白色
200	C8		H	白色

ASCII码		按 键	图形	输出颜色
十进制	十六进制			
201	C9		I	白色
202	CA		J	白色
203	CB		K	白色
204	CC		L	白色
205	CD		M	白色
206	CE		N	白色
207	CF		O	白色
208	D0		P	白色
209	D1		Q	白色
210	D2		R	白色
211	D3		S	白色
212	D4		T	白色
213	D5		U	白色
214	D6		V	白色
215	D7		W	白色
216	D8		X	白色
217	D9		Y	白色
218	DA		Z	白色
219	DB		[	白色
220	DC		\	白色
221	DD		]	白色
222	DE		↑	白色
223	DF		—	白色
224	E0		0	黄色

ASCII码		按　键	图形	输出颜色
十进制	十六进制			
225	E1		1	黄色
226	E2		2	黄色
227	E3		3	黄色
228	E4		4	黄色
229	E5		5	黄色
230	E6		6	黄色
231	E7		7	黄色
232	E8		8	黄色
233	E9		9	黄色
234	EA		0	黄色
235	EB		▒	黄色
236	EC		▒	洋红色
237	ED		▒	红色
238	EE		[illegible]	白色
239	EF		⇦	白色
240	F0		[illegible]	白色
241	F1		⇨	白色
242	F2		♪	白色
243	F3		⇑	白色
244	F4		C	白色
245	F5		O	白色
246	F6		M	白色
247	F7		X	白色
248	F8		—	黄色

ASCII码		按键	图形	输出颜色
十进制	十六进制			
249	F9		C	白色
250	FA		O	黄色
251	FB		M	洋红
252	FC		X	红色
253	FD		—	黄色
254	FE		©	白色
255	FF		♦	洋红色

特殊字符

ASCII码		输出用途	读键盘
十进制	十六进制		
128	80	光标向上	
129	81	光标向右	
130	82	光标向下	
131	83	光标向左	
132	84	CR	
133	85	清全屏幕	
134	86		DEL键
135	87		CTRL-S
136	88		向上键
137	89		向右键
138	8A		向下键
139	8B		向左键
140	8C		CTRL-R

特殊字符

ASCII码		输出用途	读键盘
十进制	十六进制		
141	8D		CTRL-C
142	8E		
143	8F		

特殊键功能

按　　键	功　能
CTRL—C	输入行无效
CTRL—I	光标向上
CTRL—J	光标向左
CTRL—K	光标向右
CTRL—M	光标向下
CTRL—R	重复最后行
CTRL—S	停止编辑

附 录 G

键盘字符定义表

字符	定 义 码
!	C8 C8 C8 C8 C8 08 C8 00 00
"	D4 D4 00 00 00 00 00 00 00
#	D4 FE D4 D4 FE D4 D4 00 00
$	C8 DE E8 DC CA FC C8 00 00
%	F2 F2 C4 C8 D0 E6 E6 00 00
&	00 D0 E8 E8 D0 EA E4 DA 00
,	C8 C8 C0 C0 C0 C0 C0 00 00
(	C2 C4 C8 C8 C8 C4 C2 00 00
)	E0 D0 C8 C8 C8 DO E0 00 00
*	C8 EA DC C8 DC EA C8 00 00
+	00 C8 C8 FE C8 C8 00 00 00
,	C0 00 C0 00 C0 C8 C8 D0 00
−	00 00 00 FE 00 00 00 00 00
·	C0 C0 C0 C0 C0 C0 C8 C0 00
/	00 C2 C4 C8 D0 E0 00 00 00
0	DC E2 E6 EA F2 E2 DC 00 00
1	C8 D8 C8 C8 C8 C8 DC 00 00
2	DC E2 E2 CC D0 E0 FE 00 00

3	DC	E2	C2	CC	C2	E2	DC	00	00
4	C4	CC	D4	E4	FE	C4	C4	00	00
5	FE	FO	FC	C2	C2	E2	DC	00	00
6	CC	DO	EO	FC	E2	E2	DC	00	00
7	FE	C2	C4	C8	D0	D0	D0	00	00
8	DC	E2	E2	DC	E2	E2	DC	00	00
9	DC	E2	E2	DE	C2	C4	C8	00	000
:	C0	C0	C2	C0	C8	C0	C0	00	00
;	C0	C0	C8	C0	C0	C8	C8	D0	00
<	C4	C8	D0	E0	D0	C8	C4	00	00
=	C0	C0	FE	C0	FE	C0	C0	00	00
>	E0	D0	C8	C4	C8	D0	E0	00	00
?	DC	E2	C4	C8	C8	C0	C8	00	00
■	FF	FF	FF	FF	FF	FF	FF	FF	00
A	C8	DC	E2	E2	FE	E2	E2	00	00
B	FC	D2	D2	DC	D2	D2	FC	00	00
C	DC	E2	E0	E0	E0	E2	DC	00	00
D	F8	E4	E2	E2	E2	E4	F8	00	00
E	FE	E0	E0	FC	E0	E0	FE	00	00
F	FE	E0	E0	FC	E0	E0	E0	00	00
G	DC	E0	E0	EE	E2	E2	DC	00	00
H	E2	E2	E2	FE	E2	E2	E2	00	00
I	DC	C8	C8	C8	C8	C8	DC	00	00
J	CE	C4	C4	C4	C4	E4	D8	00	00
K	E2	E4	E8	F0	E8	E4	E2	00	00
L	E0	E0	E0	E0	E0	E0	FE	00	00

M	E2	F6	EA	EA	E2	E2	E2	00	00
N	E2	E2	F2	EA	E6	E2	E2	00	00
O	DC	E2	E2	E2	E2	E2	DC	00	00
P	FC	E2	E2	FC	E0	E0	E0	00	00
Q	DC	E2	E2	E2	EA	E4	DA	00	00
R	FC	E2	E2	FC	E8	E4	E2	00	00
S	DC	E2	E0	DC	C2	E2	DC	00	00
T	FE	C8	C8	C8	C8	C8	C8	00	00
U	E2	E2	E2	E2	E2	E2	DC	00	00
V	E2	E2	E2	D4	D4	C8	C8	00	00
W	E2	E2	E2	EA	EA	F6	E2	00	00
X	E2	E2	D4	C8	D4	E2	E2	00	00
Y	E2	E2	D4	C8	C8	C8	C8	00	00
Z	FE	C2	C4	C8	D0	E0	FE	00	00
[	CE	C8	C8	C8	C8	C8	CE	00	00
\	C0	E0	D0	C8	C4	C2	C0	00	00
]	DC	C4	C4	C4	C4	C4	DC	00	00
↑	C8	DC	FE	C8	C8	C8	C8	00	00

附　录　H

COMX—PC 1　错误信息表

00　由用户引起的中断
01　在ASC或LEN函数中有语法错误
02　数组无说明或溢出
03　维数错误
04　DEFINT结尾不合法
05　自变量中缺括号
06　自变量溢出
07　混合形式计算
08　0做除数或对负数求对数
09　遇到不可执行的函数
10　EXIT指令必须用于FOR／NEXT 或 GOSUB／RETURN语句中
11　FOR／NEXT嵌套太多或立即执行
12　FOR语句中有语法错误
13　GOSUB嵌套太多
14　在十六进制数中有不能接受的字符
15　浮点数转换整数溢出或整数乘法溢出
16　在条件语句中有不可接受的字符
17　INPUT或FVAL不能立即执行

18　READ语句中缺变量名
19　READ或INPUT中有语法错误
20　LEN函数中有语法错误
21　赋值语句中有语法错误
22　缺少引号
23　LIST中有语法错误
24　在程序库中无此字
25　MID$函数中有语法错误
26　NEXT语句中有不接受的变量名
27　缺所需要的数或字
28　缺少括号
29　POKE语句中自变量个数不对
30　PRINT语句最后的字符不能接受
31　DATA语句中有语法错误
32　DATA语句中数据不够
33　INPUT语句中找不到相应的字符串
34　赋值语句中缺等号
35　字符串数组中缺括号
36　USR或CALL中自变量太多
37　CHR$函数有语法错误
38　二进制数中有不可接受的字符
39　行缓冲器溢出
40　输入时文件未打开
41　输出时文件未打开
42　行的结尾不可接受或语句不可执行
43　堆栈溢出

44　数的位数太多
45　在数据区有不可接受的字符
46　无此行号
47　IF语句中有非法运算符
48　存储器溢出
49　MOD语句中自变量个数不对
50　程序太长
51　自变量超出范围
52　自变量个数错误
53　自变量个数错误
54　字符串变量无定义
55　读带错误
56　写带错误
57　文件不是BASIC程序
58　文件不是BASIC数据
59　保留
60　保留
61　保留
62　未送ROM或ROM卡片
63　无足够的存储单元供重新编行号
64　重编行号设置有错误
65　无说明的下标变量
66　字符串超出127个字符
67　COLOR 中自变量的个数多于 1 个
68　SCREEN 中自变量的个数多于 1 个
69　CTONE 中自变量的个数多于 1 个

70 VOLUME 中自变量的个数多于 1 个

71 NOISE 中自变量的个数不是 2 个

72 TONE 中自变量的个数不是 3 个

附　录　I

机器语言程序简介

§1　机器语言程序编写规则

在BASIC语言中调用机器语言程序使用指令CALL。CALL通常有以下三种格式：

CALL（表达式1）

CALL（表达式1，表达式2）

CALL（表达式1，表达式2，表达式3）通过CALL指令，可以执行机器语言程序。

编写机器语言程序应牢记以下规则。

一、进入子程序的程序计数器是R2。

二、用D5（SEP R5）指令返回BASIC。

三、机器语言程序可随便使用R8，RA，RC，RD和RE。如果要使用其他寄存器，首先应存到堆栈并在返回BASIC之前复原。

注意，不可使用标准的调用和返回方式（SCRT），因为它会破坏寄存器F（RF.1）的上部。

四、COMX BASIC已建立了调用和返回的规定，不需要再用机器语言子程序重复。

编写机器语言程序必须了解1802的指令系统。

§ 2　1802A 简　介

关于1802A的详细资料请参阅1802技术说明。这里只介绍1802A的寄存器和指令系统。

一、1802A寄存器

1．D寄存器

D是数据寄存器（Accumulator），它是一个 8 位寄存器。

2．DF寄存器

DF为数据标志寄存器。它是 1 位寄存器。

3．B寄存器

B为 8 位辅助寄存器。

4．R寄存器组

R为一个寄存器组，他们由16个寄存器组成。每个寄存器都是16位的。

5．P寄存器

P是做为程序计数器使用的 4 位寄存器。

6．X寄存器

X是一个 4 位的数据指针寄存器。

7．N、I寄存器

N为低位指令寄存器，I为高位指令寄存器。它们均为4位寄存器。

8．T寄存器

T是一个8 位寄存器，它的作用是当CPU执行中断后，保留X、P寄存器内容，此时X为高位寄存器。

9．IE和Q

IE和Q都是1位寄存器，IE是中断寄存器，而Q为触发输出寄存器。

二、1802指令系统

1802指令系统由下面的指令表列出，其中操作码为十六进制数。所有寄存器的位数均是从低位（LSB）到高位（MSB）。

R（W）

M表示寄存器，这里W为N或X或P。

R（W）.O　　表示R（W）的低位

R（W）.1　　表示R（W）的高位

操作符号

M（R（N））→D；R（N）+1→R（N）

意思就是：将R（N）定义的存储器中的数送入D寄存器并且将R（N）增加1再送入R（N）。

指令系统表

指令	助记符	操作码	操作	
取N	LDN	0N	M(R（N）)→D	N不为0
取N且增	LDA	4N	M(R（N)）→D R(N)+1→R(N)	
取X	LDX	F0	M(R（(X)）→D	
取X且增	LDXA	72	M(R(X))→D； R(X)+1→R(X)	
直接取	LDI	F8	M(R(P))→D； R(P)+1→R（P）	

存N	STR	5N	D→M (R(N))
存X且减	STXD	73	D→M(R(X)); R(X)－1→R(X)
增1(N)	INC	1N	R(N)＋1→R(N)
减1(N)	DEC	2N	R(N)－1→R(N)
增1(X)	IRX	60	R(X)＋1→R(X)
得低位N	GLO	8N	R(N).0→D
放低位N	PLO	AN	D→R(N).0
得高位N	GHI	9N	R(N).1→D
放高位N	PHI	BN	D→R (N) .1
或	OR	F1	M(R (X)) ORD→D
直接或	ORI	F9	M(R (P)) ORD→D R(P) ＋1→R (P)
异或	XOR	F3	M(R (X)) XOR D→D
直接异或	XRI	FB	M(R (P)) XOR D→D R(P) ＋1→R (P)
和	AND	F2	M(R ((X)) AND D→D
直接和	ANI	FA	M(R(P)) AND D→D R(P)＋1→R (P)
变换右移	SHR	F6	D右移 LSB (D) →DF 0→MSB (D)
进位 变换右移	SHRC RSHR	76	D右移LSB(D)→DF, DF, DF→MSB (D)
变换左移	SHL	FE	D左移 MSB (D) →DF,

			0→LSB (D)
变换左移进位	SHLC RSHL	7E	D左移 MSB (D) →DF, DF→LSB (D)
加	ADD	F4	M(R (X)) +D→DF, D
直接加	ADI	FC	M(R (P)) +D→DF, R (P) +1→R (P)
进位加	ADC	74	M(R(X))+D+DF→DF, D
直接进位加	ADCI	7C	M (R(P))+D+DF→DF, D; R(P) +1→R (P)
减D	SD	F5	M(R(X)) -D→DF, D
直接减D	SDI	FD	M(R (X)) -D→DF, D; R(P) +1→R (P)
借位减D	SDB	75	M(R(X))-D-(非DF)→DF,D
直接借位减D	SDBI	7D	M(R(P))-D-(非DF) →DF, D; R (P) +1→R (P)
减存储器	SM	F7	D-M(R(X))→DF, D
直接减存储器	SMI	FF	D-M(R(P))→DF, D; R (P) +1→R (P)
借位减存储器	SMB	77	D-M(R(X))—(非DF) →DF, D
借位直接减存储器	SMBI	7F	D—M(R(P)) —(非DF) →DF, D; R (P) +1→R (P)
短转移	BR	30	M(R(P))→R(P).0

非短转移	NBR	38	R(P)+1→R(P)
如果D=0则短转移	BZ	32	IF D=0, M(R(P))→R(P).0 ELSE R(P)+1→R(P)
如果D非零则短转移	BNZ	3A	IF D非0, M(R(P))→R(P).0 ELSE R(P)+1→R(P)
如果DF=1则短转移	BDF	33	IF DF=1, M(R(P))→R(P).0 ELSE R(P)+1→R(P)
正或0	BPZ		
如果DF=0则短转移	BNF	3B	IF DF=0, M(R(P))→R(P).0 ELSE R(P)+1→R(P)
为负	BM		
如果Q=1则短转移	BQ	31	IF Q=1, M(R(P))→R(P).0 ELSE R(P)+1→R(P)
如果Q=0则短转移	BNQ	39	IF Q=0, M(R(P))→R(P).0 ELSE R(P)+1→R(P)
如果EF1=1则短转移	B1	34	IF EF1=1, M(R(P))→R(P).0 ELSE R(P)+1→R(P)

如果EF1＝0则短转移	BN1	3C	IF EF1＝0, M(R(P))→R(P).0 ELSE R(P)+1→R(P)
如果EF2＝1则短转移	B2	35	IF EF2＝1, M(R(P))→R(P).0 ELSE R(P)+1→R(P)
如果EF2＝0则短转移	BN2	3D	IF EF2＝0, M(R(P))→R(P).0 ELSE R(P)+1→R(P)
如果EF3＝1则短转移	B3	36	IF EF3＝1, M(R(P))→R(P).0 ELSE R(P)+1→R(P)
如果EF3＝0则短转移	BN3	3E	IF EF3＝0, M(R(P))→R(P).0 ELSE R(P)+1→R(P)
如果EF4＝1则短转移	B4	37	IF EF4＝1, M(R(P))→R(P).0 ELSE R(P)+1→R(P)
如果EF4＝0则短转移	BN4	3F	IF EF4＝0, M(R(P))→R(P).0 ELSE R(P)+1→R(P)

长转移	LBR	CO	M(R(P))→R(P).1 M(R(P)+1)→ R(P).0
非长转移	NLBR	C8	R(P)+2→R(P)
如果D=0 则长转移	LBZ	C2	IF D=0,M(R(P)) →R(P).1 M(R(P)+1)→R (P).0 ELSE R(P)+2→ R(P)
如果D非0 则长转移	LBNZ	CA	IF D非0 M(R(P) →R(P).1 M(R(P)+1)→R(P) .0ELSE R(P)+2 →R(P)
如果DF=1 则长转移	LBDF	C3	IF DF=1, M(R (P))→R(P).1 M(R(P)+1)→R (P).0 ELSE R(P)+3→ R(P)
如果DF=0 则长转移	LBNF	CB	IF DF=0,M(R (P))→R(P).1 M(R(P)+1)→ R(P).0

			ELSE R(P)+2→R(P)
如果Q=1则长转移	LBQ	C1	IF Q=1, M(R(P))→R(P).1 M(R(P)+1)→R(P).0 ELSE R(P)+2→R(P)
如果Q=0则长转移	LBNQ	C9	IF Q=0, M(R(P))→R(P).1 M(R(P)+1)→R(P).0 ELSE R(P)+2→R(P)
短跳跃	SKP	38	R(P)+1→R(P)
长跳跃	LSKP	C8	R(P)+2→R(P)
如果D=0则长跳跃	LSZ	CE	IF D=0, R(P)+2→R(P) 否则继续
如果D非0则长跳跃	LSNZ	C6	IF D非0, R(P)+2→R(P) 否则继续
如果DF=1则长跳跃	LSDF	CF	IF DF=1, R(P)+2→R(P) 否则继续
如果DF=0	LSNF	C7	IF DF=0, R(P)

			+ 2 → R(P)
则长跳跃			否则继续
如果Q = 1	LSQ	CD	IF Q = 1, R(P) + 2 → R (P)
则长跳跃			否则继续
如果Q = 0	LSNQ	C5	IF Q = 0, R(P) + 2 → R(P)
则长跳跃			否则继续
如果IE = 1	LSIE	CC	IF IE = 1, R(P) + 2 → R (P)
则长跳跃			否则继续
空转	IDL	0 0	等待DMA或中断：M(R(0)) → BUS
不操作	NOP	C4	继续
转P	SEP	DN	N → P
转X	SEX	EN	N → X
转Q	SEQ	7B	1 → Q
重建Q	REQ	7A	0 → Q
存	SAV	78	T → M (R (X))
放X，P至堆栈	MARK	79	(X,P) → T；(X,P) → M(R(2)) 然后P → X；R (2) − 1 → R(2)
转回	RET	70	M(R(X)) → (X, P)；R(X) + 1 → R (X)；1 → IE

截止	DIS	71	M(R(X))→(X, P)； R(X)+1→R(X)； 0→IE
输出 1	OUT1	61	M(R(X))→BUS； R(X)+1→R(X)； N线为1
输出 2	OUT2	62	M(R(X))→BUS R(X)+1→R(X)；N 线为2
输出 3	OUT3	63	M(R(X))→BUS R(X)+1→R(X)； N线为3
输出 4	OUT4	64	M(R(X))→BUS R(X)+1→R(X)；N 线为4
输出 5	OUT5	65	M(R(X))→BUS R(X)+1→R(X)；N 线为5
输出 6	OUT6	66	M(R(X))→BUS R(X)+1→R(X)；N 线为6
输出 7	OUT7	67	M(R(X))→BUS R(X)+1→R(X)； N线为7
输入 1	INP1	69	BUS→M(R(X)) BUS→D；N线为1

输入 2	INP2	6A	BUS→M（R（X）） BUS→D；N线为 2
输入 3	INP3	6B	BUS→M（R（X）） BUS→D；N线为 3
输入 4	INP4	6C	BUS→M（R（X）） BUS→D；N线 为4
输入 5	INP5	6D	BUS→M（R（X）） BUS→D；N线 为 5
输入 6	INP6	6E	BUS→M（R（X）） BUS→D；N线为 6
输入 7	INP7	6F	BUS→M（R（X）） BUS→D；N线为7

三、1802 操作码表

MSD \ LSD	0	1	2	3	4	5	6	7
0	IDLE	LDN						
1	INC							
2	DEC							
3	BR	BQ	BZ	BDF BPZ BGE	B1	B2	B3	B4
4	LDA							
5	STR							
6	IRX	OUT 1	OUT 2	OUT 3	OUT 4	OUT 5	OUT 6	OUT 7
7	RET	DIS	LDXA	STXD	ADC	SDB	SHRC RSHR	SMB
8	GLO							
9	GHI							
A	PLO							
B	PHI							
C	LBR	LBQ	LBZ	LBDF	NOP	LSNQ	LSNZ	LSNF
D	SEP							
E	SEX							
F	LDX	OR	AND	XOR	ADD	SD	SHR	SM
	0	1	2	3	4	5	6	7

8	9	A	B	C	D	E	F	
								0
								1
								2
SKP NBR	BNQ	BNZ	BNF BM BL	BN1	BN2	BN3	BN4	3
								4
								5
	INP1	INP2	INP3	INP4	INP5	INP6	INP7	6
SAV	MARK	REQ	SEQ	ADCI	SDBI	SHLC RSHL	SMBI	7
								8
								9
								A
								B
LSKP NLBR	LBNQ	LBNZ	LBNF	LSIE	LSQ	LSZ	LSDF	C
								D
								E
LDI	ORI	ANI	XRI	ADI	SDI	SHL	SMI	F
8	9	A	B	C	D	E	F	

附录 J

COMX—BASIC 语句和函数索引

LEN
LET
LIST
LOG
MEM
MID$
MOD
MUSIC
NEW
NEXT
NOISE
PEEK
PI
PLOAD
POKE
PR
PRINT

REM
RENUMBER
RESTORE
RETURN
RND
RUN
RUN
SCREEN
SGN
SHAPE
SIN
SQR
TAB
TIME
TIMOUT
TONE
TRACE
USR
VOLUME
WAIT

封面设计：辛　奇

新书目号：102-144
统一书号：15244·0024

定　　价：　1.50　元

COMX—PC1电脑实验与指导